Universidad
Carlos III de Madrid

Interconexión de redes de sensores inalámbricos 802.15.4 en localizaciones remotas

Autor: Rodrigo Muñoz Castejón
Tutor: Ángel Cuevas Rumín
Co-Tutor: Manuel Ureña

Fecha de Presentación

Marzo 2011

Componentes del tribunal:

- Presidente: Celeste Campos
- Secretario: Ricardo Romeral
- Vocal: Marcelino Lázaro

Agradecimientos:

En primer lugar me gustaría dar las gracias a Ángel por haberme dirigido este proyecto de fin de carrera. Por su confianza, disponibilidad y apoyo durante todo el proyecto, sería un placer poder coincidir en algún proyecto común en el futuro.

También me gustaría agradecer a Manolo las dudas resueltas en los momentos claves del proyecto.

A todos mis compañeros y amigos de la universidad, porque sin todos vosotros, seguro que no estaba escribiendo estas líneas.

A mis compañeros de trabajo que me han ayudado a dar ese último empujón al proyecto.

A mis amigos de siempre que me han animado a terminar la carrera, y que siempre están cuando los necesito.

Y finalmente, no puedo dejar de agradecer la comprensión y el apoyo de mis padres, hermanos y el resto de familiares, puesto que ellos han sido los que más me han tenido que aguantar a lo largo de toda la carrera y durante la elaboración de este proyecto.

A todos muchas gracias.

Resumen del proyecto

En este proyecto se propone una solución eficaz para comunicar redes de sensores inalámbricas 802.15.4, situadas en localizaciones geográficas remotas, utilizando redes TCP/IP.

Presentando así la posibilidad de comunicar redes de sensores muy distantes de forma completamente transparente para los dispositivos finales.

El desarrollo del proyecto incluye el estudio de las características y prestaciones de las tecnologías para este tipo de redes inalámbricas. Argumentando el uso de las tecnologías utilizadas.

En el proyecto se ha incluido la implementación de un escenario práctico, que demuestra la comunicación de los dispositivos de forma remota. Además este documento incluye un tutorial que describe la puesta en marcha del escenario de ejemplo.

1 Tabla de contenido

2 Introducción 7
2.1 Prólogo 7
2.2 Redes de área local inalámbrica (WLAN) 8
2.3 Redes de sensores inalámbricas (WSN) 8
2.3.1 Tipos de Sensores 10
2.3.2 Aplicaciones y ejemplos de redes de sensores inalámbricas (WSN) 12
2.4 Redes de sensores y actuadores inalámbricas (WSAN) 15
3 Motivación y Objetivos 16
3.1 Enfoque 16
3.2 Propuesta 17
3.3 Beneficios de la solución propuesta 18
4 Estado del arte 20
4.1 WiFi 802.11 21
4.2 Bluetooth 22
4.3 Wireless USB 23
5 El Estándar 802.15.4 24
5.1 Aspectos generales 29
5.2 Red de área personal inalámbrica (WPAN) 30
5.3 Beacons 31
5.4 Identificadores y estructura de la trama 32
5.5 Protocolos de nivel superior que utilizan 802.15.4 36
5.5.1 ZigBee 36
5.5.2 WirelessHART 39
6 Solución Propuesta 40

6.1 Objetivos 40
6.2 Trama UC3M 42
6.3 Comunicación entre coordinador y PC 44
6.4 PC-Bridge 44
6.5 Servidor de direcciones 45
7 Despliegue del escenario implementado 47
7.1 Introducción. 47
7.2 Presentación del Hardware. 49
7.3 Módulos y clases principales. 51
7.3.1 Endpoint. 52
7.3.2 Coordinador 61
7.3.3 PC como punto de acceso 802.15.4 72
7.3.4 Servidor de direcciones 80
7.4 Aplicación de la propuesta en un escenario real 83
8 Presupuesto 86
8.1 Costes de investigación y desarrollo. 86
8.2 Costes Material 87
8.3 Coste Total del proyecto 87
8.4 Presupuesto solución domótica 88
8.4.1 Presupuesto solución UC3M 88
8.4.2 Presupuesto solución AMX 89
8.4.3 Conclusiones Presupuestos domótica. 90
9 Tutorial de despliegue del ejemplo 91
10 Conclusiones 95
10.1 Evaluación de resultados 95
10.2 Trabajos futuros 96
11 Referencias 97

2 Introducción

2.1 Prólogo

Las redes de sensores han surgido en la última década como nuevo concepto en adquisición y tratamiento de datos con múltiples aplicaciones en distintos campos tales como entornos industriales, domótica, entornos militares, detección ambiental. Podríamos definir una red de sensores inalámbricos (WSN, *Wireless Sensor Network)* como un conjunto de elementos autónomos (nodos) interconectados de manera inalámbrica, cuyas principales premisas se enfocan en:

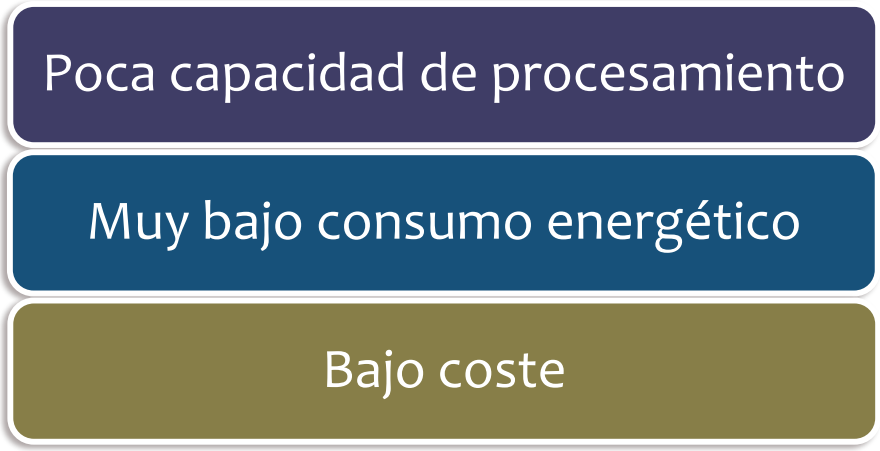


Ilustración 1: Perspectiva de las redes de sensores

2.2 Redes de área local inalámbrica (WLAN)

Es una red que cubre un área equivalente a la red local de una empresa, con un alcance aproximado de cien metros. Permite que las terminales que se encuentran dentro del área de cobertura puedan conectarse entre sí. Existen varios tipos de tecnologías de este tipo, las cuales esta sujetas a las necesidades básicas de dichas redes, como son principalmente:

- La cobertura
- El tiempo de conexión
- El ancho de banda
- Consumo energético

2.3 Redes de sensores inalámbricas (WSN)

WSN es el acrónimo de *Wireless Sensor Network*, esto es, una Red de Sensores Inalámbricos, donde cada nodo de dicha red recibe el nombre de mota "*mote*", dispositivo compuesto de un microprocesador con memoria, sensor/es, una radio de baja potencia y una batería.

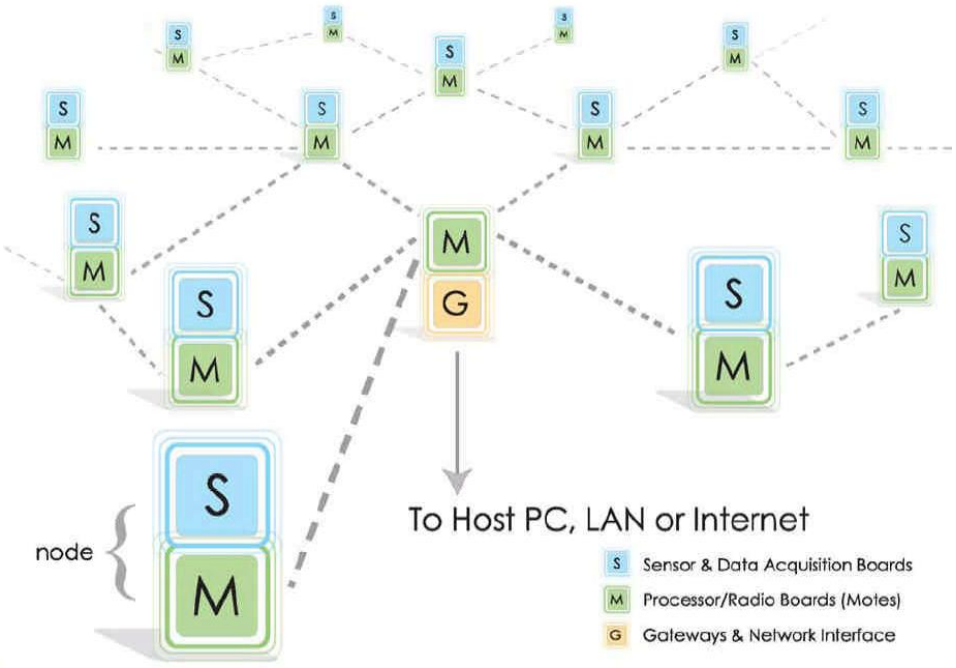


Ilustración 2

Hoy en día el mercado presenta una gran variedad de hardware que se ajusta a las necesidades de las redes. Los dispositivos pueden variar mucho de unas redes a otras, tanto es así que podemos encontrar desde dispositivos con potentes microprocesadores de dimensiones razonables, hasta absolutamente diminutos como los conocidos "Smart dust", por el momento se han conseguido desarrollar sensores inalámbricos, denominados "Spec", que contienen chips miniaturizados a menos de un milímetro cúbico.

Ilustración 3: Ejemplo smart dust

Fuente de la Imagen:
http://ambientintelligence.wordpress.com

Ejemplo de dispositivo del tipo "Smart Dust" (polvo inteligente) para la monitorización de la humedad en el cultivo

2.3.1 Tipos de Sensores

Hoy en día existe una gran variedad de sensores, tan diversos como prácticamente todos los fenómenos físicos que se deseen medir. Todos ellos ofrecen unas grandes posibilidades a las redes de sensores inalámbricas, extendiendo cada vez más el abanico de escenarios potenciales. A continuación mostramos una clasificación de los sensores más utilizados en la actualidad. [1]

Detectores de ultrasonidos

- Los detectores de ultrasonidos resuelven los problemas de detección de objetos de prácticamente cualquier material, normalmente se usan para control de presencia/ausencia, distancia o rastreo.

Interruptores manuales

- Estos son los sensores más básicos, incluye pulsadores, llaves, selectores rotativos y conmutadores de enclavamiento.

Productos infrarrojos

- Los componentes optoelectrónicos son sensores fiables y económicos. Se incluyen diodos emisores de infrarrojos (IREDs), sensores y montajes.

Sensores de caudal de aire

- Los sensores de caudal de aire contienen una estructura de película fina aislada térmicamente, que contiene elementos sensibles de temperatura y calor.

Sensores de corriente

- Los sensores de corriente monitorizan corriente continua o alterna, pueden ser utilizados como un elemento de respuesta para controlar un motor o regular la cantidad de trabajo que realiza una máquina.

Sensores de efecto Hall

- Son semiconductores y por su costo no están muy difundidos pero en codificadores ("encoders") de servomecanismos se emplean mucho.

Sensores de temperatura

- Estos sensores consisten en una fina película de resistencia variable con la temperatura.

Sensores de humedad

- Los sensores de humedad relativa/temperatura y humedad relativa contienen un elemento sensible capacitivo en base de polímeros que interacciona con electrodos de platino.

Sensores de posición de estado sólido

- Los sensores de posición de estado sólido, detectores de proximidad de metales y de corriente.

Sensores de turbidez

- Los sensores de turbidez aportan una información rápida y práctica de la cantidad relativa de sólidos suspendidos en el agua u otros líquidos.

Sensores magnéticos

- Entre las aplicaciones se incluyen brújulas, control remoto de vehículos, detección de vehículos, realidad virtual, sensores de posición, sistemas de seguridad e instrumentación médica.

Interruptores final de carrera

- El microswitch es un conmutador de 2 posiciones con retorno a la posición de reposo, básicamente diferencian dos estados, muy utilizados en detectores de nivel de fluidos .

Sensores de presión y fuerza

- Las aplicaciones afines a estos productos incluyen instrumentos para aviación, laboratorios, controles de quemadores y calderas, comprobación de motores, tratamiento de aguas residuales y sistemas de frenado.

Sensores de químicos

- Aplicables a todo tipo de sondas, como por ejemplo PH, O2, Salinidad…

2.3.2 Aplicaciones y ejemplos de redes de sensores inalámbricas (WSN)

Las nuevas tecnologías están abriendo las puertas cada vez más a las WSNs, capaces de procesar enormes cantidades de datos. Los sensores son cada día más habituales en nuestro entorno, y todavía podrían dar mucho más de sí. Así lo cree toda una industria tecnológica que está detrás de ellos, siendo cada vez más las empresas y los equipos de investigadores que trabajan en el desarrollo de este tipo de dispositivos.

A continuación se muestran algunos de los ejemplos en aplicaciones desarrolladas a día de hoy [2].

2.3.2.1 *Agricultura y ganadería*

Combinando sensores como humedad, temperatura y luminosidad se pueden detectar riesgo de heladas, posibles enfermedades de las plantas o la necesidad de riego según el nivel de humedad de la tierra, entre otras.

Ilustración 4

Es posible monitorizar la temperatura a la que se encuentran las crías para mantenerla en los niveles adecuados o incluso controlar el nivel de estrés de los animales monitorizando la agitación del rebaño con sensores de vibración y movimiento.

Ilustración 5

2.3.2.2 *Medio Ambiente*

A través de una red de sensores inalámbrica se pueden monitorizar especies en extinción, o detectar y prevenir incendios forestales, además en ciudades se monitoriza los niveles de contaminación de la atmósfera y recogen datos sobre el clima, con el fin de mantener un desarrollo sostenible.

Ilustración 6

2.3.2.3 *Salud*

Hospitales, centros de Salud e incluso entornos cercanos a pacientes con limitaciones o enfermedades disponen cada vez mas de redes de sensores que desarrollan mediciones biométricas.

Incluso centros de alto rendimiento utilizan redes de sensores inalámbricos para monitorizar deportistas en el terreno de juego, de esta forma, entrenadores y médicos pueden evaluar de una mejor forma las respuestas de los deportistas.

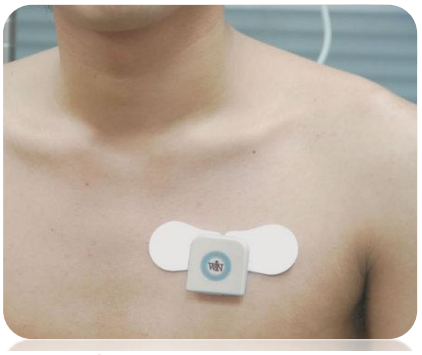


Ilustración 7

2.3.2.4 *Procesos Industriales*

En este campo las aplicaciones están muy extendidas y son muy variadas, el control de las emisiones y escapes de gases es un aspecto que preocupa tanto empresarial como ecológicamente para el desarrollo sostenible.

Aparte de esto las redes de sensores también sirven para mejorar los procesos de fabricación o las condiciones de mantenimiento.

Ilustración 8

2.3.2.5 *Seguridad y Emergencias*

Las aplicaciones de las redes de sensores en este campo son innumerables, permiten crear entornos más seguros y gestionar alarmas.

También son muy utilizadas para la prevención, ayuda y gestión en situaciones de emergencia o desastres naturales.

Ilustración 9

2.3.2.6 *Otras aplicaciones*

Por poner algunos ejemplos más podemos mencionar redes de sensores en vehículos (coches, aviones, trenes…), también se están integrando redes en grandes ciudades para controlar el tráfico, logística de transporte, marketing, lectura de contadores, etc.

2.4 Redes de sensores y actuadores inalámbricas (WSAN)

En las Redes Inalámbricas de Sensores y Actuadores (WSAN, Wireless Sensor and Actor Networks) además de nodos sensores existen nodos actuadores.

Los sensores van reuniendo información sobre el medio físico, mientras que los actuadores toman decisiones y ejecutan las acciones apropiadas sobre el entorno.

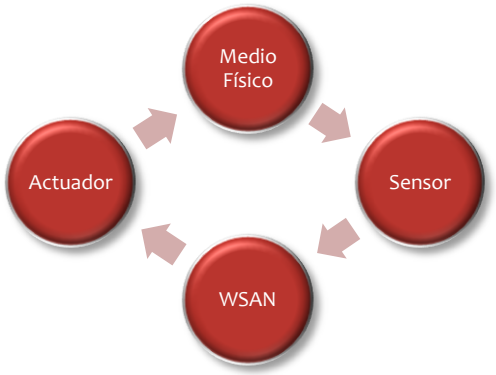


Diagrama 1: Correspondencia de las redes de sensores con el medio físico.

Este tipo de redes tienen algunas características que no son propias de las WSN:

- Mientras que los sensores son dispositivos pequeños, baratos, con capacidad de comunicación y procesamiento limitados; los actuadores habitualmente consumen más energía, y son mas caros.
- La capacidad de reacción de los actuadores ante un evento en tiempo real es una cuestión importante. Es necesario introducir mecanismos de coordinación entre los sensores y los actuadores.
- El número de sensores suele ser muy superior al número de actuadores, aunque por supuesto depende del escenario concreto.

Es precisamente en este tipo de redes donde se motiva este proyecto, tomando aun mas sentido, pues presenta la posibilidad de operar de forma remota desde escenarios distantes y diversos.

3 Motivación y Objetivos

3.1 Enfoque

En ocasiones existe la necesidad de comunicar dos sensores completamente independientes, puede ser el caso por ejemplo, de un sensor que necesite transmitir cierta información a un actuador el cual no pertenece a su propia red.

Esto no es posible cuando la separación de los dispositivos excede los márgenes de cobertura de las tecnologías inalámbricas usadas en las redes de sensores, por lo tanto, necesitamos una solución que nos permita esta comunicación. Con este proyecto proponemos una solución sencilla, que resulta completamente transparente a los nodos finales, delegando toda la complejidad del enrutamiento a dispositivos mucho más potentes.

De esta forma además podría establecerse una organización jerárquica en árbol de las subredes, donde los nodos o dispositivos finales que forman las hojas del árbol pueden estar miles de kilómetros distantes.

La siguiente ilustracion muestra un ejemplo donde 3 redes de sensores A-B-C, situadas en localizaciones geotraficas remotas, etablecen una comunicación entre el dispositivo A.2 y el B.1.

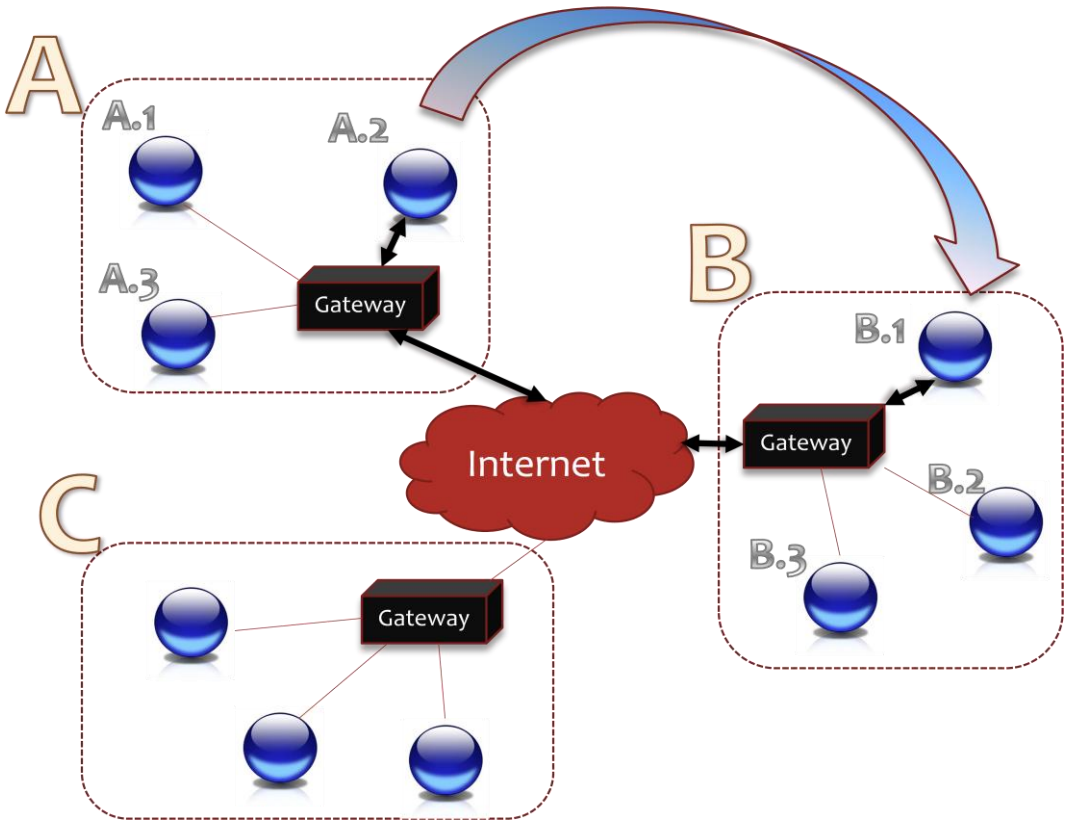


Diagrama 2: Ejemplo explicativo de comunicación

3.2 Propuesta

La propuesta de este proyecto consiste técnicamente, en establecer una comunicación entre WSNs remotas. Para ello planteamos una nueva cabecera UC3M con la información de direccionamiento necesaria para encaminar las tramas generadas por los dispositivos.

Conjuntamente es necesario integrar e implementar equipos conectados a internet, capaces de procesar las tramas procedentes de las WSNs.

3.3 Beneficios de la solución propuesta

Nuestra solución representa la base para nuevas aplicaciones, desplegando una abanico de posibilidades para servicios de WSN o WSAN.

En primer lugar, a nivel de prestaciones, podemos aumentar prácticamente ilimitadamente la red de sensores/actuadores anidando diferentes subredes, de forma transparente para los dispositivos finales. Este punto es una de las principales motivaciones del proyecto.

A nivel de funcionalidad las bondades de la solución propuesta se basan en primer lugar en la interoperatibilidad entre redes muy distantes ampliando su rango de acción. En segundo lugar, el despliegue necesario para la interconexión presenta un escenario idóneo para el control sincronizado de las subredes. En último lugar, se amplía el concepto del simple sensor/actuador, de forma que se pueden desarrollar aplicaciones más complejas sin necesidad de protocolos adicionales, ya que para la interconexión se utiliza la tecnologia estandar TCP/IP, enormemente extendida.

Veamos todo esto de una forma esquematizada:

Ampliar el radio de Acción

- La intercomunicación de dos o mas subredes con la misma funcionalidad, nos ofrece la posibilidad de ampliar geográficamente la superficie de acción.
- Además , se pueden cubrir grandes areas de forma selectiva, sin necesidad de puntos de interconexion o repetidores de señal, situando subredes en puntos estrategicos.

Se extiende el principio Accion/reaccion

- La interconexión de este tipo de redes de forma jerárquica, permite acciones simultaneas y en cadena de dispositivos, por ejemplo, el valor de un sensor puede suponer la acción de varios actuadores simultáneamente, o de forma secuencial, sincronizando su funcionamiento de manera sencilla.

4 Estado del arte

Llegados a este punto, debemos plantear las distintas tecnologías inalámbricas que podrían ser utilizadas en nuestra WSAN, el objetivo es escoger la más adecuada para cumplir las expectativas del proyecto, el cual reúne las siguientes pretensiones principales:

Consumo energético→ En toda WSN la energía pasa a ser un punto crucial ya que sus dispositivos están alimentados por baterías.

Limitaciones hardware→ Se cuenta con una capacidad de proceso limitada en cada dispositivo, lo cual hace indispensable que el hardware sea lo más sencillo posible, así como su transceptor radio.

Topología Dinámica→ En una red de sensores, la topología siempre es cambiante y éstos tienen que adaptarse para poder comunicar nuevos datos adquiridos.

Además de estas necesidades principales, debemos valorar algunas otras como son el tiempo de conexión de nuevos dispositivos, el alcance efectivo de las transmisiones, el número de dispositivos que forman la red.

A continuación describiremos muy brevemente algunas de las tecnologías inalámbricas que podrían usarse en las redes de sensores, y describiremos en profundidad 802.15.4 que es la tecnología que vamos a utilizar, de forma que pueda apreciarse la diferencia entre estas para poder valorar la tecnología utilizada en el proyecto.

4.1 WiFi 802.11

Es una marca de la *Wi-Fi Alliance,* la organización comercial que adopta, prueba y certifica que los equipos cumplen los estándares *802.11.*

La norma IEEE 802.11 esta enormemente extendida mundialmente, periódicamente se realizan reajustes en el protocolo que aumentan el ancho de banda, optimizan las conexiones y en general aumentan las prestaciones.

Inicialmente grandes compañías de desarrollo e implementación de WSN como CrossBow utilizaron esta tecnología en sus primeros dispositivos, pero fueron remplazados rápidamente por nuevas tecnologías más orientadas a mecanismos de bajo consumo.

Los dispositivos de este tipo de red normalmente están conectados a la red eléctrica, o requieren de carga de sus baterías cada poco tiempo, esto es debido a su consumo energético, tanto en transmisión/recepción como en du proceso de funcionamiento.

El nuevo mercado de los "smartphones" y "Tablet Pc", dispositivos portables y con capacidad energética limitada, a supuesto que tanto las tarjetas de red, como las últimas versiones de 802.11, estén centrándose en reducir el consumo energético en los dispositivos y los tiempos de conexión a la red, aun así estas mejoras no son suficientes cuando comparamos con otras tecnologías tal y como veremos más adelante.

Ilustración 10: Dispositivos con comunicación WiFi

En conclusión, las redes WiFi ofrecen unas prestaciones que podríamos considerar de alto nivel, por lo que este tipo de redes inalámbricas es utilizado básicamente en la conexión entre ordenadores, o incluso periféricos que requieran grandes anchos de banda como podría ser una cámara IP o algunos Smartphone.

4.2 Bluetooth

Bluetooth es el nombre comercial de la especificación industrial IEEE 802.15.1, que define un estándar global de comunicación inalámbrica que posibilita la transmisión de voz y datos.

La especificación de Bluetooth define un canal de comunicación mediante un enlace por radiofrecuencia en la banda de los 2,4 GHz, un dispositivo debe implementar alguno de los perfiles Bluetooth, estos definen el uso del canal Bluetooth.

Fue diseñado desde sus orígenes para la transmisión de voz, por lo que es estrictamente necesario vincular los dispositivos antes de su comunicación.

Bluetooth es posiblemente el más extendido de de los protocolos de este tipo, su evolución desde la versión v1.1 (1994) hasta la actual v3.0 ha sido enorme, se ha aumentado la velocidad e incrementado el rango de acción, además el salto a la versión v2.1 supuso una disminución de consumo casi 5 veces menor que las versiones anteriores. Si a todo esto le sumamos sus transceptores de bajo coste parece el candidato perfecto para desarrollar redes del tipo WSN.

La última versión de Bluetooth v3.0 se ha desarrollado especialmente para trabajar en colaboración con WiFi, lo que facilitaría enormemente la implementación del propósito de este proyecto, sin embargo, aunque Bluetooth es altamente eficiente, su consumo energético es mayor si lo comparamos con otros estándares (802.15.4), este consumo proviene de algunos factores esenciales:

1) Bluetooth ofrece una calidad de voz (Voice Quality - Enhanced Voice Processing) especialmente orientado a teléfonos móviles, esta capacidad requiere de un consumo de energía, el cual pequeños dispositivos, como podría ser un sencillo sensor de presencia no requieren.
2) Otros protocolos (802.15.4), especialmente orientado a redes de sensores de muy bajo consumo, desarrollan sistemas en los que los dispositivos permanecen en un estado de letargo la mayor parte del tiempo, permitiendo un gran ahorro de energía.
3) El tamaño de la trama Bluetooth es bastante grande y poco flexible en comparación con otros protocoles, en este tipo de redes, la transmisión y recepción de paquetes supone la mayor parte del consumo energético.

En conclusión, Bluetooth ofrece unas prestaciones que lo hacen candidato al desarrollo de WSN, el gran inconveniente de esta tecnología sería evidente en redes de sensores que realizan tareas de monitorización, donde es necesario desplegar los sensores por largos periodos de tiempo.

4.3 Wireless USB

Este protocolo fue lanzado en sus inicios con gran expectación, aunque no se ha extendido demasiado en el mercado, destaca por su gran ancho de banda en distancias muy cortas que combina la sencillez de uso de USB con la versatilidad de las redes inalámbricas, puede lograr tasas de transmisión de hasta 480 Mbps en rangos de tres metros, su uso está enfocado a mandos para juegos, cámaras de video y fotos, discos duros externos, etc.

Obviamente las características de este servicio no se ajustan a una WSN como las enfocadas en el proyecto.

5 El Estándar 802.15.4

Es un estándar que define el nivel físico y el control de acceso al medio de redes WLAN con tasas bajas de transmisión de datos, desplegando redes con muy bajo consumo. La eficiencia energética de este protocolo reside fundamentalmente en el uso de las tramas "Beacon", que permiten sincronizar los dispositivos de la red para que puedan permanecer en modo ahorro de energía el mayor tiempo posible, esto supone una gran ventaja para el desarrollo WSN que realicen tanto tareas de monitorización como de control.

Este estándar sirve de base para otras especificaciones como ZigBee o WirelessHART, que mencionaremos más adelante, cuyo propósito es ofrecer una solución completa para este tipo de redes definiendo los niveles superiores de la pila de protocolos que el estándar no cubre.

A continuación, se muestran un conjunto de tablas y diagramas que comparan las características entre 802.15.4 y el resto de tecnologías mencionadas previamente.

	Wireless USB	**WiFi**	**Bluetooth**	**802.15.4**
Frecuencia	3.1-10.6 GHz	2.5-5 GHz	2.4GHz	2.4GHz 868/915GHz
Ancho de Banda	480Mbps(3m) 110Mbps(10m)	11-108 Mbps	1-3 Mbps	20/40/250 Kbps
Cobertura	3-10 m	20-250 m	1-100 m	1-75 m

Tabla 1: Comparativo de tecnologías inalámbricas

En la tabla vemos de forma esquematizada algunas de las caracteristicas de las redes inalambricas. Podemos ver como 802.15.4, puede utilizar tres bandas de frecuencia, en un principio la banda de 868 MHz esta disponible en europa, 915MHz en USA y 2400 MHz a nivel mundial, con sus correspondientes anchos de banda, de hecho un dispositivo que implementa el 802.15.4 puede transmitir en una de tres posibles bandas de frecuencia, por lo que cada vez existe menor diferenciacion.

La tabla presenta un alcance maximo de 75m para 802.15.4, esta medida se pondera en condiciones óptimas las cuales es dificil que se presenten en escenarios reales, en nuestras pruebas en el laboratorio hemos podido experimentar fallos en la conexión en distancias superiores a 25m. Esto depende enormemente de la ganancia de los transceptores, ademas de factores como campos electromagneticos, materiales de construccion de edificio,etc.

Esta limitación en la distancia podría ser minimizada de dos formas:

- La primera es utilizando tecnologias como zigbee que permiten por ejemplo multisalto entre los dispositivos, esto permitiría cubrir distancias razonables.
- La segunda opción es una de las que nos concierne en este proyecto, interconectando puntos de acceso TCP/IP que permitan ampliar el radio de accion.

La primera opción es descartada cuando las WSNs están muy distantes, pues exiten limitaciones en el numero de saltos entre los dispositivos, por otra parte, lógicamente existen sobrecostes si contemplamos la opción de distribuir numerosos dispositivos.

En Contraposición la segunda opción planteada como propuesta del proyecto nos permite conectar dispositivos tan distantes como sea necesario, sin tener para ello grandes sobrecostes.

Este diagrama muestra los rangos de funcionamiento de las distintas tecnologias en funcion de la distancia y la capacidad.

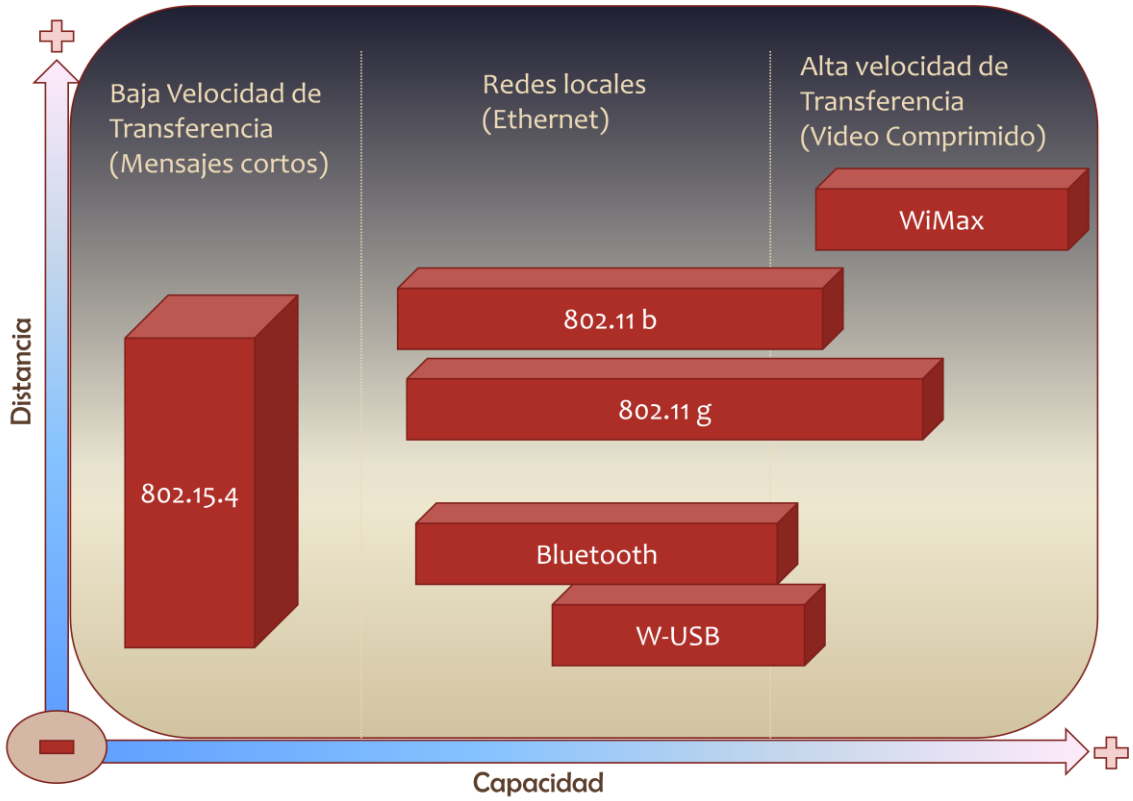


Ilustración 11: Comparativo de tecnologías inalámbricas referente a la cobertura y la tasa de transmisión.

Como podemos ver 802.15.4 tiene una tasa de transferencia relativamente baja, debido a dicha tasa, 802.15.4 es una tecnología aplicable a WSNs, que tienen requerimientos moderados de ancho de banda. Sin embargo 802.15.4 no sería una tecnología adecuada para aplicaciones que exigen una alta tasa de transferencia, como puede ser el video.

A continuación se muestran dos graficas que representan el consumo energético de los dispositivos. Estos valores son orientativos y no deben tomarse como referencias exactas pues dependen mucho tanto del software como del hardware de cada dispositivo.

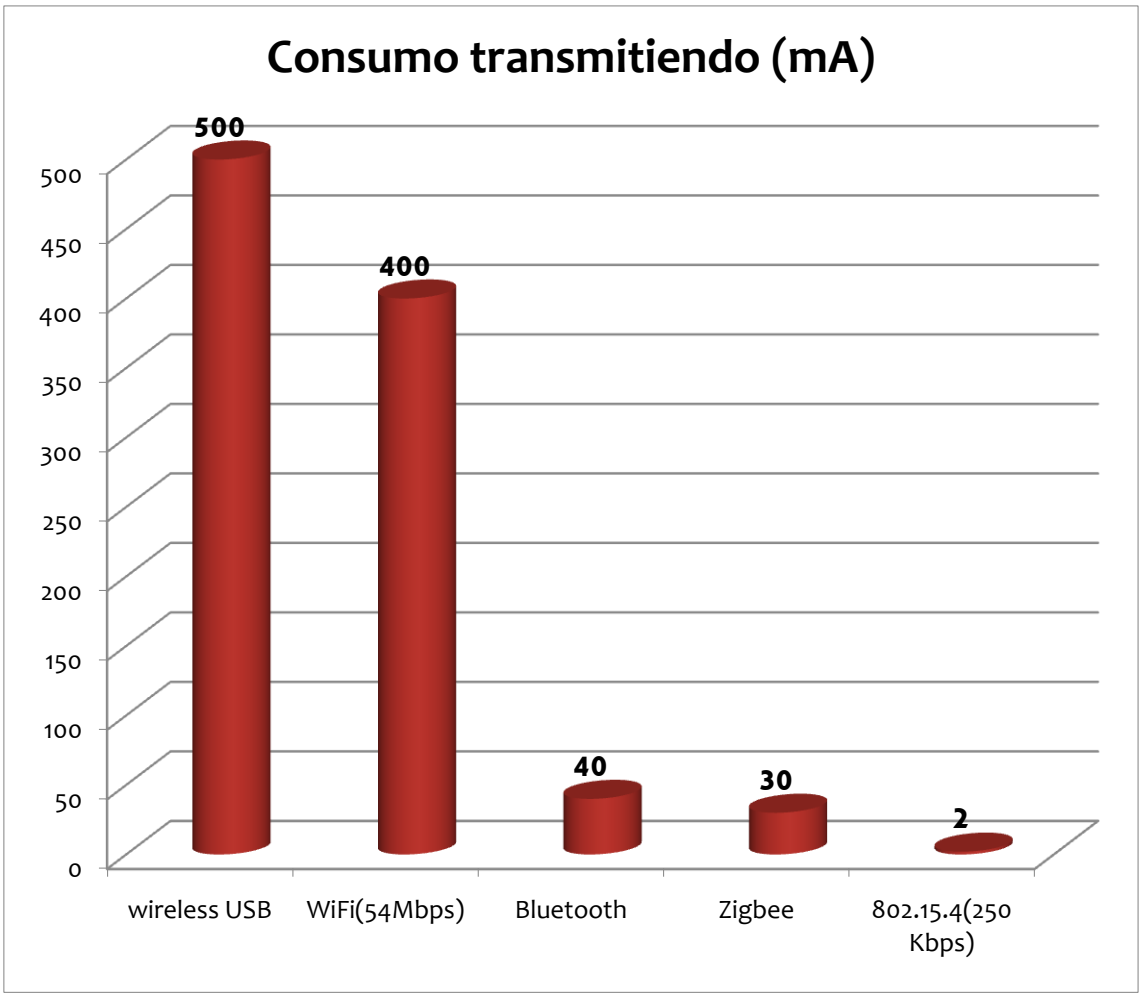


Ilustración 12: Consumo transmitiendo de las tecnologías inalámbricas

El consumo energético en un dispositivo es directamente proporcional al tamaño de los paquetes y la cantidad de datos transmitidos, aún así es evidente la eficiencia de 802.15.4.

ZigBee es una tecnología que complementa 802.15.4 a nivel de red y aplicación, hablaremos más adelante acerca de esta tecnología. Es importante para el proyecto mencionar la significativa diferencia entre ZigBee y 802.15.4, esta diferencia se debe a las cabeceras adicionales de ZigBee, y a la mayor complejidad del protocolo que exige un mayor tiempo de procesado de los datos.

A continuación una grafica comparativa de consumo en reposo.

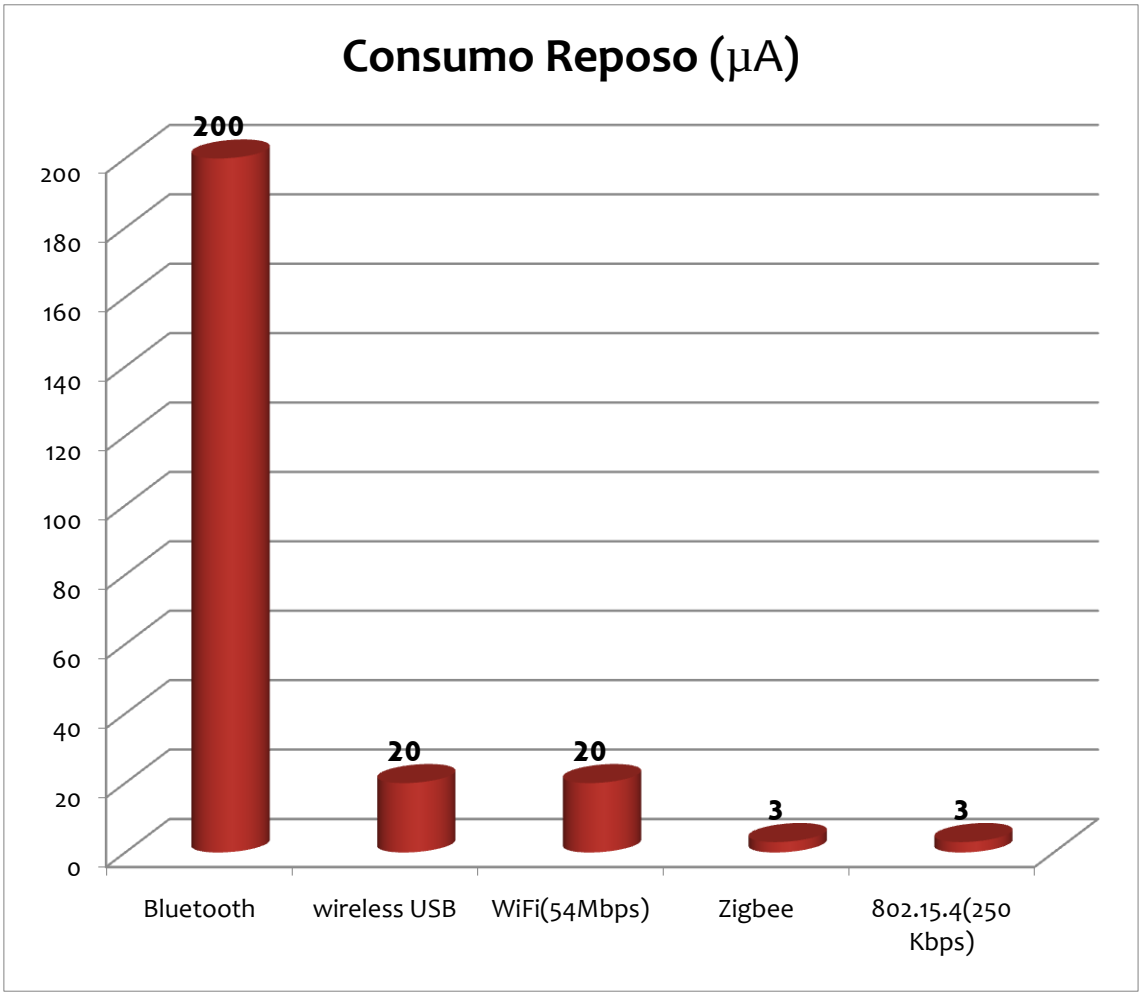


Ilustración 13: consumo en reposo de las tecnologías inalámbricas

En esta grafica hay que mencionar que el alto consumo de Bluetooth en reposo se debe a que esta tecnología nunca está en auténtico reposo, los dispositivos se encuentran vinculados y sincronizados.

Podemos apreciar también los mínimos valores en ZigBee y 802.15.4, por supuesto el consumo es el mismo pues ZigBee trabaja sobre 802.15.4 donde ambos mantienen las mismas rutinas en reposo.

5.1 Aspectos generales

El estándar (802.15.4) define dos tipos de nodo en la red:

Dispositivo de funcionalidad completa (*full-function device*, FFD). Implementa un modelo general de comunicación que le permite establecer un intercambio con cualquier otro dispositivo.

Puede funcionar como coordinador de una red de área personal (PAN), en este caso será responsable de su red, sincronizando sus nodos mediante el envío de Beacons.

Puede actuar también como un nodo repetidor, o bien simplemente como un dispositivo final de la WSN.

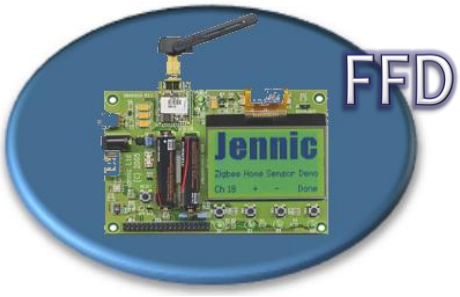


Ilustración 14: Dispositivo de funcionalidad completa

Dispositivos de funcionalidad reducida (*reduced-function device*, RFD). Se plantean como dispositivos muy sencillos con recursos y necesidades de comunicación muy limitadas. Por ello, sólo pueden comunicarse con FFD y nunca pueden ser coordinadores.

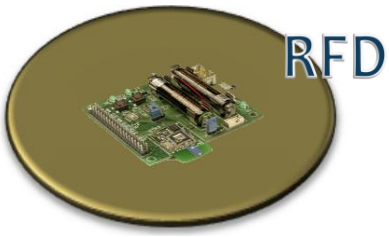


Ilustración 15: dispositivo de funcionalidad reducida

Las redes de nodos pueden construirse como redes punto a punto o en estrella, siendo la implementación más habitual esta última, en la que el coordinador va a ser siempre el nodo central. El estándar no define un nivel de red, por lo que no se soportan funciones de ruteo de forma directa, que pueden ser proporcionadas

por otros protocolos (Eje. ZigBee), así pueden realizarse comunicaciones en varios saltos, brindado la posibilidad de topologías más complejas. En cualquier caso, toda red necesita al menos un FFD que actúe como su coordinador, así las redes están compuestas por grupos de dispositivos separados por distancias adecuadas.

Pueden imponerse otras restricciones topológicas, en concreto, el estándar menciona el árbol de clúster, como una estructura que aprovecha que los RFD sólo pueden conectarse con un FFD al tiempo para formar redes en las que los RFD son siempre hojas del árbol.

En este proyecto se implementará una topología en estrella de forma que cada uno de los paquetes que circule por la red pasa por el nodo coordinador, de esta forma, aunque sacrificamos algunas bondades de topologías como la de malla, logramos una gran robustez en el sistema al delegar toda la responsabilidad en el FFD.

5.2 Red de área personal inalámbrica (WPAN)

Las redes inalámbricas de área personal, WPAN, por sus siglas en Ingles, Wireless Personal Area Network, son redes que comúnmente cubren distancias del orden de los 10 m como máximo, normalmente utilizadas para conectar varios dispositivos portátiles personales sin la necesidad de utilizar cables.

Las redes inalámbricas relativas a nuestro proyecto se sitúan dentro de este tipo de redes, donde se utiliza un identificador (PAN ID) para determinar cada una de estas, la longitud de este depende del protocolo en concreto, en nuestro caso (802.15.4) es de 2 bytes.

Este identificador que denominamos PAN_ID o PID, es establecido por el máximo responsable de la WPAN, el coordinador. Todos los dispositivos de una WPAN comparten el mismo PAN_ID, de esta forma evitamos que se interfieran otros coordinadores cercanos.

La solución propuesta en este proyecto, añade un significado global al PAN_ID. Ofreciendo la posibilidad de interconectar WPANs remotas, para ello se incluyen los PAN_ID dentro de la nueva cabecera UC3M.

5.3 Beacons

La motivación clave para el uso de esta tecnología reside principalmente en el notable ahorro energético de los dispositivos finales, esto es en parte gracias a las tramas de "Beacon" que permiten mantener latentes a los dispositivos la mayor parte del tiempo. Por lo tanto, la principal funcionalidad de las conocidas como ¨Beacon Frames¨ es permitir a los dispositivos despertar solamente cuando es transmitida esta trama de guía.

Las tramas de guía (Beacon) son enviadas por el coordinador en intervalos definidos, delimitan lo que se denomina como Superframe, estos intervalos, definidos por el coordinador, pueden ser tan cortos como 15 mseg o tan largos como 245 seg, en cualquier El tiempo entre cada uno de ellos se divide en 16 "time slots" idénticos.

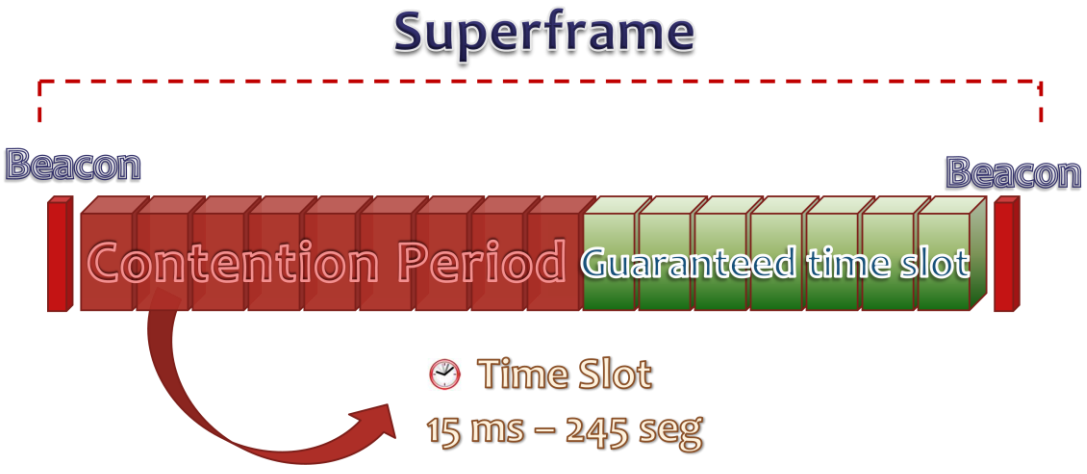


Ilustración 16: estructura de una Superframe

En la Superframe se diferencian dos periodos en los que los dispositivos sincronizados en la WPAN pueden transmitir:

- Periodo de Contención: Si un dispositivo desea transmitir, puede acceder al medio mediante CSMA-CA, pero debe terminar su transmisión antes de que termine el "time Slot".

- Periodo Garantizado: El resto de los "slots" están reservados por el coordinador de la red para garantizar el acceso a dispositivos determinados.

Las tramas de Beacon son utilizadas igualmente para sincronización de nuevos dispositivos o nodos caídos, de tal forma que si un nuevo dispositivo tiene intención de solicitar al ingreso a una red 802.15.4 debe esperar una nueva trama Beacon que le permita solicitar su admisión.

5.4 Identificadores y estructura de la trama

En el estándar (802.15.4) cada dispositivo posee un identificador único de 64 bits (MAC Address), aunque habitualmente, y si se dan ciertas condiciones en el entorno, pueden utilizarse identificadores cortos de 16 bits, éstos tienen sentido únicamente dentro del dominio de cada PAN.

En el proyecto, cada coordinador es responsable del direccionamiento de su PAN, manteniendo un registro de cada uno de los dispositivos sincronizados, estos pueden utilizar en todo momento ambos identificadores Short Address 16bits, Long Address 64 bits.

Una de los puntos críticos del proyecto es la estructura de las tramas. Como se mencionó en la introducción estas tramas se verán complementadas con una nueva cabecera suplementaria UC3M, con el fin de añadir información a los enrutadores (básicamente coordinadores de cada una de las PAN). El estándar define cuatro tipos de trama[5]:

MAC Command Frame: utilizada para manejar todo el control de entidad MAC. Es un mecanismo para el control o configuración a distancia de los dispositivos de los nodos. Permite que un director de

la red centralizado, pueda configurar a los dispositivos individualmente.

Ack Frame: usado para confirmar la recepción exitosa de la trama. Proporciona el intercambio de información activa desde el receptor al emisor de que el paquete fue recibido sin error. Esta trama aprovecha el tiempo de silencio (quiet time), especificado por la norma, inmediatamente después de la transmisión del paquete de datos.

Beacon Frame: La trama Beacon añade un nuevo nivel de funcionalidad a la WSN. Los dispositivos de los nodos pueden despertarse solamente cuando es transmitida una señal de guía o "Beacon", escuchar su dirección y si no la escuchan volver al estado de ahorro de energía.

Data Frame: trama utilizada para todas las transferencias de datos. En lo relativo al proyecto las únicas tramas que se verán modificadas en el protocolo serán las tramas de datos, por lo cual, analizaremos en profundidad este tipo.

La trama general de MAC se la denomina PPDU (Phy Protocol Data Unit), a continuación se desgrana la trama para entender mejor cada uno de sus campos y donde incidirá especialmente el nuevo protocolo UC3M

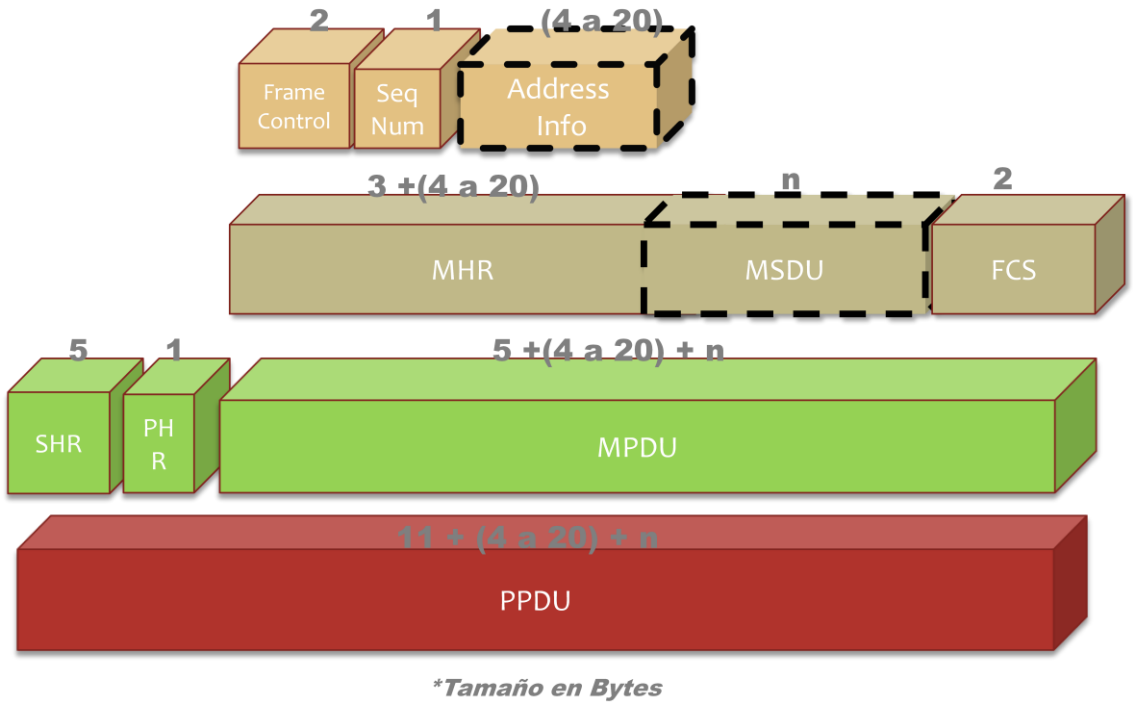


Ilustración 17: Estructura de la trama 802.15.4

PPDU

SHR→ Contiene un preámbulo de 32 bits que esta relaccionado con ajustes en la frecuencia y usos especiales, ademas se reservan 8 bits para un delimitador de comienzo de la trama.

PHR→ Se utilizan 7 bits para especificar la longitud de la carga de datos (en bytes), de forma que la longitud del paquete va de 0 a 127, normalmente los paquetes que circulan por la red tienen tamaños de 30 a 60 bytes, solo algunas aplicaciones como el control de periféricos requiere paquetes mayores.

MPDU (MAC Protocol Data Unit)

MPDU

MHR→ Encabezado que contiene información de control, su tamaño depende del tipo de direccionamiento (identificadores 32bits o 16bits).

FCS→ (Frame Check Sequency), es una trama de chequeo de 16 bits CRC.

MSDU→ Se denomina comúnmente "Payload", su longitud es variable sin superar nunca los 127bytes estipulados, este campo tendrá vital importancia pues aquí es donde introduciremos nuevas cabeceras que nos permitan direccionar paquetes fuera de la red.

MPDU

Frame Control→ indica el tipo de trama MAC que se pretende trasmitir, especifica el formato, el campo de dirección y controla los mensajes "ack.

Numero de secuencia→ verifica la integridad de la trama MAC, Una transmisión se considera exitosa solo cuando la trama de enterado (ACK) contiene la misma secuencia de números que la secuencia anterior trasmitida.

Direccionamiento→ El tamaño varía desde 0 a 20 bytes, de forma que por ejemplo la trama ACK no contiene información en este campo.

5.5 Protocolos de nivel superior que utilizan 802.15.4

Como se ha comentado, 802.15.4 es un estándar que define el nivel físico y el control de acceso al medio en redes WLAN. Existen algunas tecnologías que complementan el estándar a nivel de red y aplicación, prestando servicios adicionales o ampliando la robustez. A continuación describiremos las dos más extendidas del mercado.

5.5.1 ZigBee

ZigBee es un conjunto de protocolos de alto nivel de comunicación inalámbrica. Su objetivo son las aplicaciones que requieren comunicaciones seguras con baja tasa de envío de datos y maximización de la vida útil de sus baterías.

Fue un proyecto formado por seis promotores (Honeywell, Invensys, Mitsubishi, Motorola, Philips, y Samsung) y más de 80 participantes. El primer perfil se declaró a mediados de 2003, se definieron especificaciones globales de aplicaciones inalámbricas fiables, económicas y de baja potencia basadas en la norma IEEE 802.15.4. El siguiente diagrama muestra la organización básica:

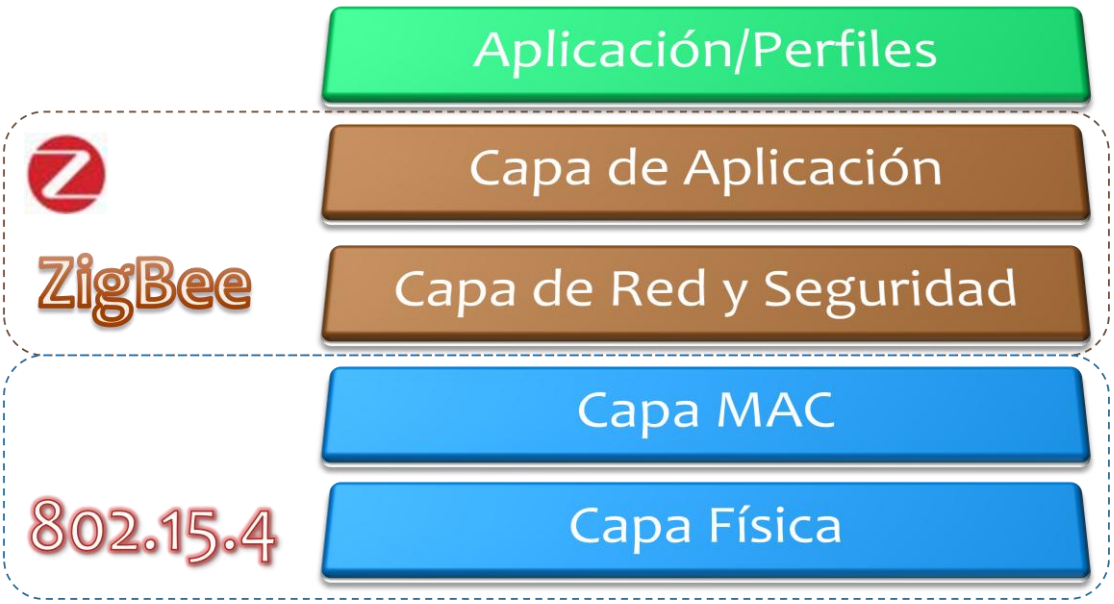


Ilustración 18: Pila de capas Zigbee

Las características básicas de ZigBee son[6]:

- Menor potencia y menor coste que otras WPAN (Como Bluetooth).
- Potencia Tx 1mW(hasta 10mW en CE, hasta 100 mW en EEUU)
- Los nodos están gran parte del tiempo "dormidos"(Larga
- duración: 2 años)
- Rango alcance: 10-100 m, hasta 400 m con 10 mW)
- Bit-rate entre los 20kB/s y 250kB/s
- Se permiten hasta un total de 65534 nodos/red
- 3 bandas comunicación: 868MHz, 915MHz, 2.4GHz

A continuación se muestran de modo informativo dos graficas comparativas acerca de ZigBee realizadas por la Universidad de Granada[3]:

BER(Bit Error Rate)→ Es el numero de Bits de un flujo de datos a través de un canal de comunicación que han sido alterados por ruido, interferencias, distorsiones o fallos de sincronización, dividido entre el número de bits totales transmitidos.

SNR(Signal-to-noise ratio)→Es el cociente de dividir la potencia de la señal entre la potencia del ruido en dB.

La siguiente grafica muestra la gran robustez frente a otras tecnologías inalámbricas.

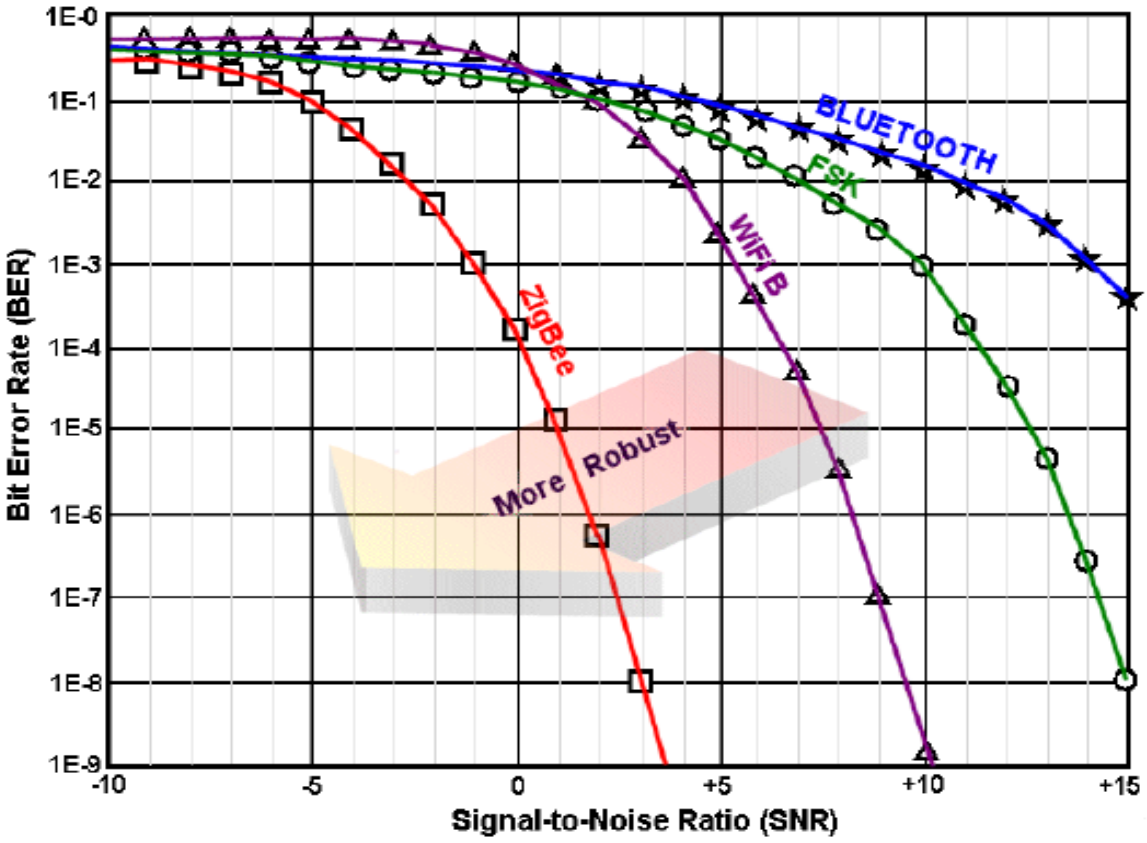


Ilustración 19: Comparativo referente a los errores en la comunicación.

El siguiente diagrama contrasta ZigBee frente a otras tecnologías inalámbricas teniendo en cuenta el alcance de la red, y la tasa de transferencia de datos.

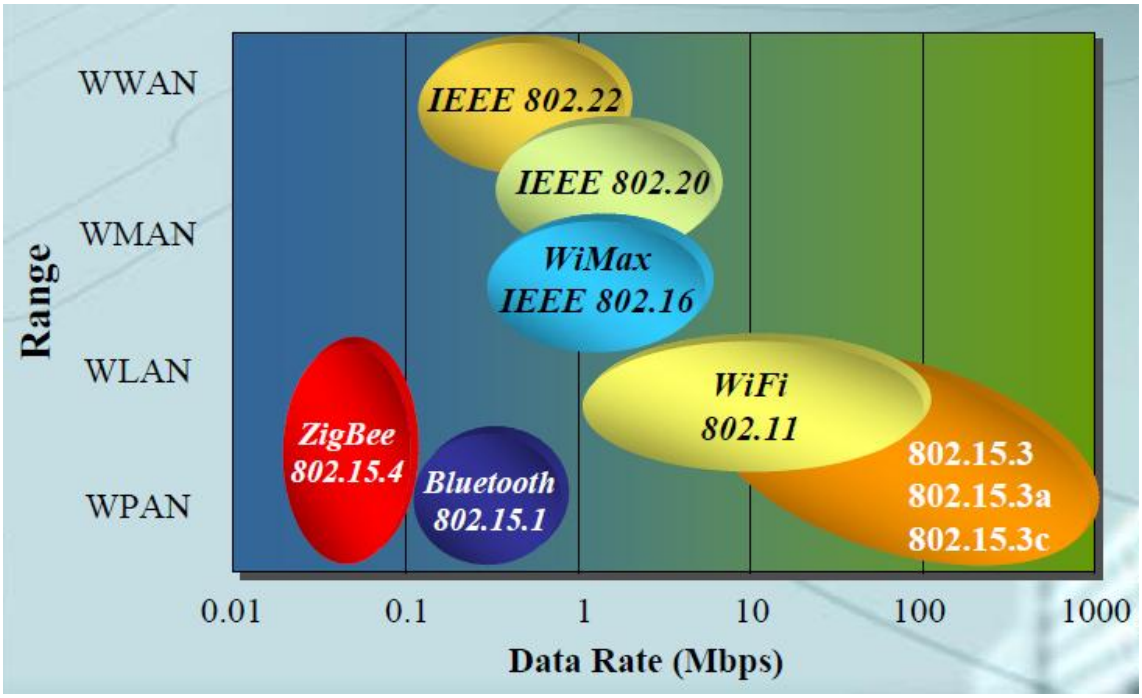


Ilustración 20: Comparativo de tecnologías inalámbricas.

5.5.2 WirelessHART

Desarrollado por *HART Communication Foundation,* Utiliza radios compatibles con IEEE 802.15.4, su principal argumento es su capacidad de auto-organización, desarrollando redes muy robustas frente a caídas de nodos y con una gran flexibilidad.

Aunque no ha tenido demasiado impacto en el mercado, es una tecnología perfecta para aplicaciones que requieran mucha robustez y fiabilidad. Está desplegada especialmente en laboratorios y sobre algunos equipos hospitalarios.

6 Solución Propuesta

6.1 Objetivos

El objetivo principal de proyecto es conseguir la interconexión de redes de sensores, situadas en localizaciones geográficas remotas a través TCP/IP, inicialmente se ha investigado el funcionamiento de la red WSN y el comportamiento de sus motas. Una vez realizado este estudio se añade una cabecera a la trama de datos para que pueda ser enviado fuera de la red de sensores a la cual pertenece.

Una visión global de todo esto:

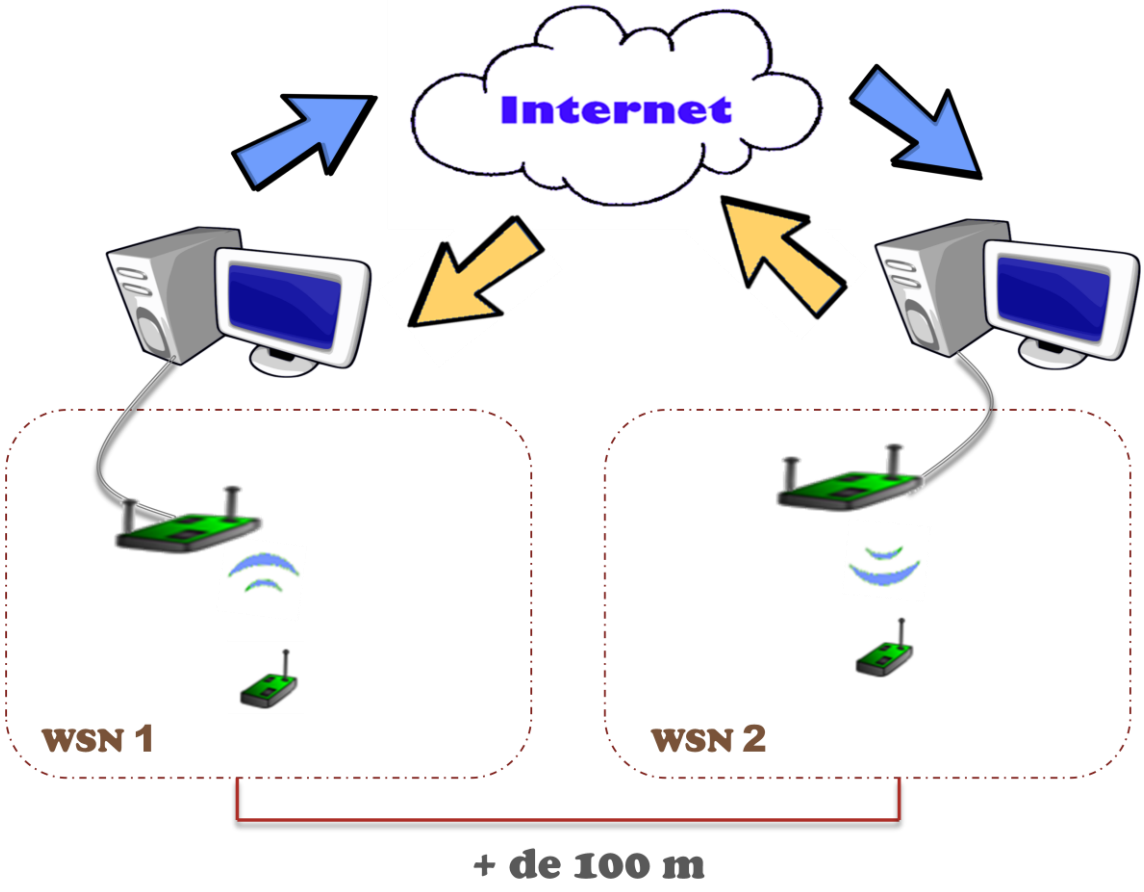


Diagrama 3: Diagrama de comunicación de la solución propuesta.

Para llevar a cabo un escenario como este, deberíamos abordar varios aspectos:

1. La estructura de las tramas de datos del estándar 802.15.4 no contienen suficiente información para determinar un direccionamiento fuera de las WSAN, por lo que habría que añadir algunas cabeceras a la trama básica.
2. El coordinador debe ser modificado para encaminar los paquetes de la subred.
3. El coordinador necesita estar conectado a algún tipo de dispositivo con comunicación TCP/IP, lo más sencillo sería un PC.
4. El PC implementa una aplicación que le permite establecer comunicación directa con el coordinador, y a su vez, establecer comunicaciones remotas con otras máquinas a través de TCP/IP.
5. En último lugar, un terminal conectado a la red desplegara las funciones de servidor de direcciones, asociando las direcciones IP de los PC con los identificadores de red 802.15.4(PAN_ID)

De esta forma, el direccionamiento de los paquetes es responsabilidad en primer lugar de los coordinadores de WSAN, y en segundo lugar de los PCs que interconectan las redes de sensores, así los dispositivos finales se mantienen al margen de cualquier direccionamiento complejo, ayudando a la portabilidad y transparencia de la solución propuesta.

Basándonos en estas metas buscamos una solución eficiente y estable al problema, que sea completamente factible en un escenario real.

6.2 Trama UC3M

Anteriormente se ha descrito la estructura de la trama 802.15.4 y la función de cada uno de los campos. Ahora pasamos a describir la cabecera UC3M añadida a la trama que nos permitirá el direccionamiento fuera de la subred.

Esta nueva cabecera va incluida en el Payload de la trama 802.15.4, (MSDU), y añade una longitud fija de 8bytes (4x2bytes) a la trama, el coordinador de la subred será el encargado de interpretar estos nuevos campos, veamos un diagrama de la estructura:

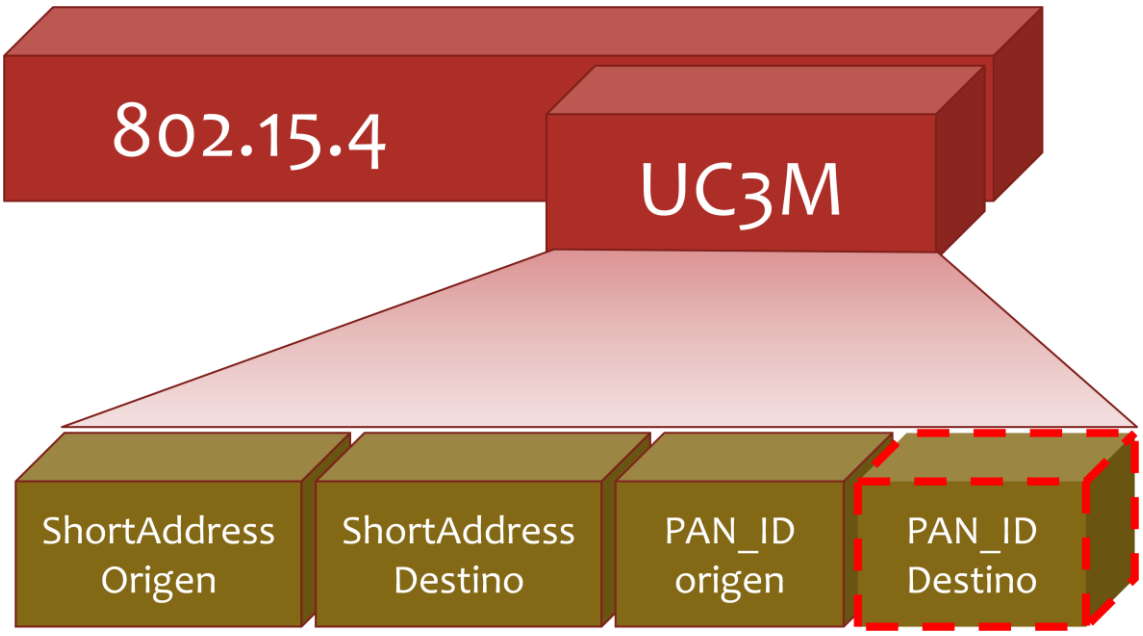


Diagrama 4: estructura de la trama UC3M

Los identificadores de WSN "PAN_ID" serán utilizados por los coordinadores para encaminar los paquetes, estos pueden reconocer fácilmente si el paquete recibido pertenece a una comunicación dentro de su propia red comparando PAN_ID origen y destino. Las direcciones origen y destino junto con los PAN_ID identifican inequívocamente a cada una de las motas de las distintas WSN.

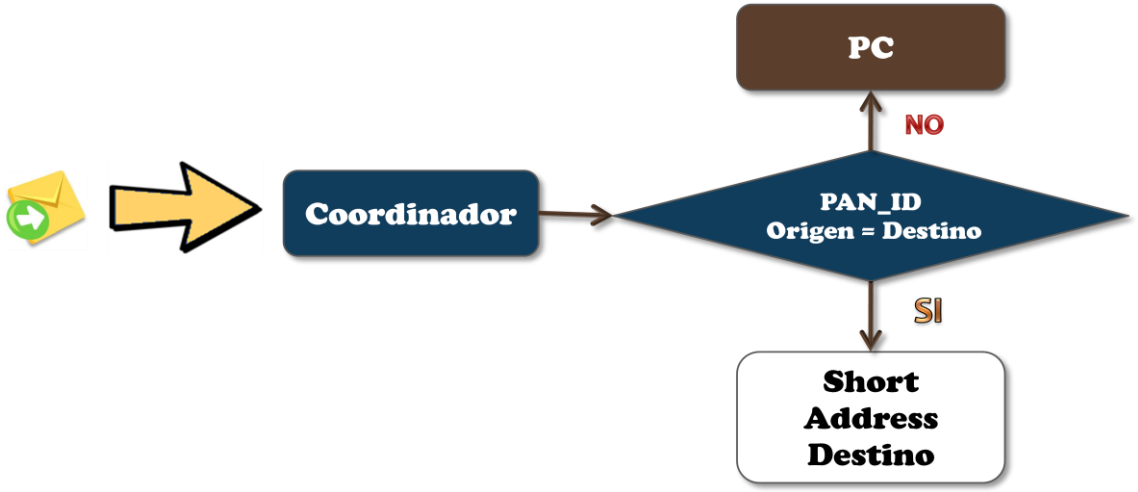


Diagrama 5: Diagrama de decisión del coordinador.

Una de las características más destacables de la solución propuesta es la transparencia para los dispositivos finales, simplificando al máximo sus funciones, y evitando modificar en lo posible su código para facilitar su portabilidad.

En contraposición, nuestra solución es viable en topologías físicas en estrella, donde todos las tramas son enviados al coordinador, siendo este el encargado del enrutamiento.

De esta forma, cada vez que el coordinador recibe una trama de datos, extrae previamente la información de la cabecera UC3M, para decidir su enrutamiento, mas adelante hablaremos acerca del proceso de enrutamiento de paquetes por parte del coordinador en la sección Coordinador.

Esto podría plantearse como un inconveniente de la propuesta pues centralizamos toda la comunicación de la WSN en el coordinador, además de establecer una topología de estrella fija, por el contrario de esta forma las motas pueden centrarse únicamente en realizar su función, esto supone un importante ahorro energético en las motas.

6.3 Comunicación entre coordinador y PC

La conexión sincronizada entre el coordinador y el PC es de gran importancia para el correcto desarrollo del proyecto. Existen diferentes formas validas de intercomunicarlos.

En nuestra solución se ha utilizado para la comunicación el puerto serie, donde el PC realiza una espera activa de los caracteres recibidos por el puerto, de esta forma se ha creado un protocolo de comunicación propio capaz de diferenciar entre las tramas de la WSN y los mensajes de monitorización de la red.

6.4 PC-Bridge

Para la interconexión de WSN a través de TCP/IP necesitamos implementar una aplicación en el PC que permita el tránsito de datos entre las dos redes WSN. Sobre estas maquinas se ejecutara el programa Java PC Bridge.

Las funciones principales de la aplicación instalada en ambos extremos:

- ✓ Lectura e interpretación de los caracteres recibidos desde el coordinador por el puerto serie.
- ✓ Envío de datagramas por el puerto serie hacia el coordinador.
- ✓ Resolución de direcciones IP<>802.15.4 (a través del servidor de direcciones).
- ✓ Reenvío de paquetes en formato TCP/IP a WSN remotas.
- ✓ Interpretación de paquetes TCP/IP con información 802.15.4 de WSN remotas.

Como se ha mencionado anteriormente, cuando hablábamos de la comunicación entre coordinador y PC, se van a desarrollar trazas de monitorización de la red, ya que no disponemos de ningún mecanismo para escanear el trafico de las redes WSN, esto nos permite tener constancia de lo que está sucediendo en cada momento en la red, y cuál es el intercambio de mensajes.

6.5 Servidor de direcciones

El servidor de direcciones es una funcionalidad adicional que se desarrollo para dotar al proyecto de mayor flexibilidad.

Cuando se empezó a trabajar en el proyecto, se presentó el problema de que para que las redes de sensores estuvieran conectadas de forma remota, cada una de estas debería almacenar la información de las direcciones IP y PANID de las redes con las cuales conectarse. Esto supone un gran inconveniente por dos motivos:

- Para un gran número de WSN interconectadas, la información duplicada en cada una de las redes puede ser un inconveniente.
- Es muy habitual el direccionamiento dinámico de las redes IP, por lo que si la dirección IP cambia en alguna de las WSN, es necesario actualizar la información en todas y cada una de las tablas.

Para solucionar esto existen algunos protocolos habitualmente utilizados en routers que automatizan estas tareas, pero optamos por la solución más sencilla y estable, un servidor de direcciones.

De esta forma se creó una aplicación (servidor de direcciones) que se encuentra en una maquina remota la cual es conocida por el resto de los equipos, así, cada WSN solo almacena una dirección IP que se presupone fija, además si una WSN cambia su dirección IP, simplemente basta con actualizar la tabla del servidor.

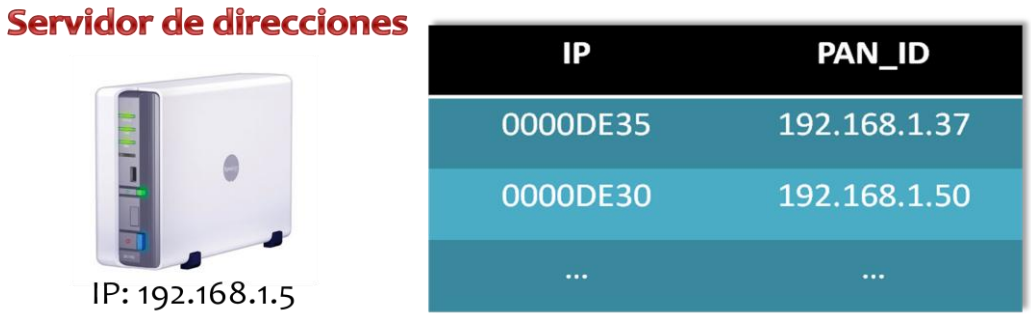


Ilustración 21: Muestra de servidor de direcciones.

Supongamos que un equipo responsable de una WSN ha recibido una solicitud de una mota para enviar un paquete a una PANID remota, simplemente basta con enviar una solicitud de resolución de direcciones al servidor, indicándole la PANID destino y la PANID origen (esta última para actualizar la tabla), el servidor le contestara enviando la IP donde se encuentra el equipo sobre el que se conecta la PANID deseada.

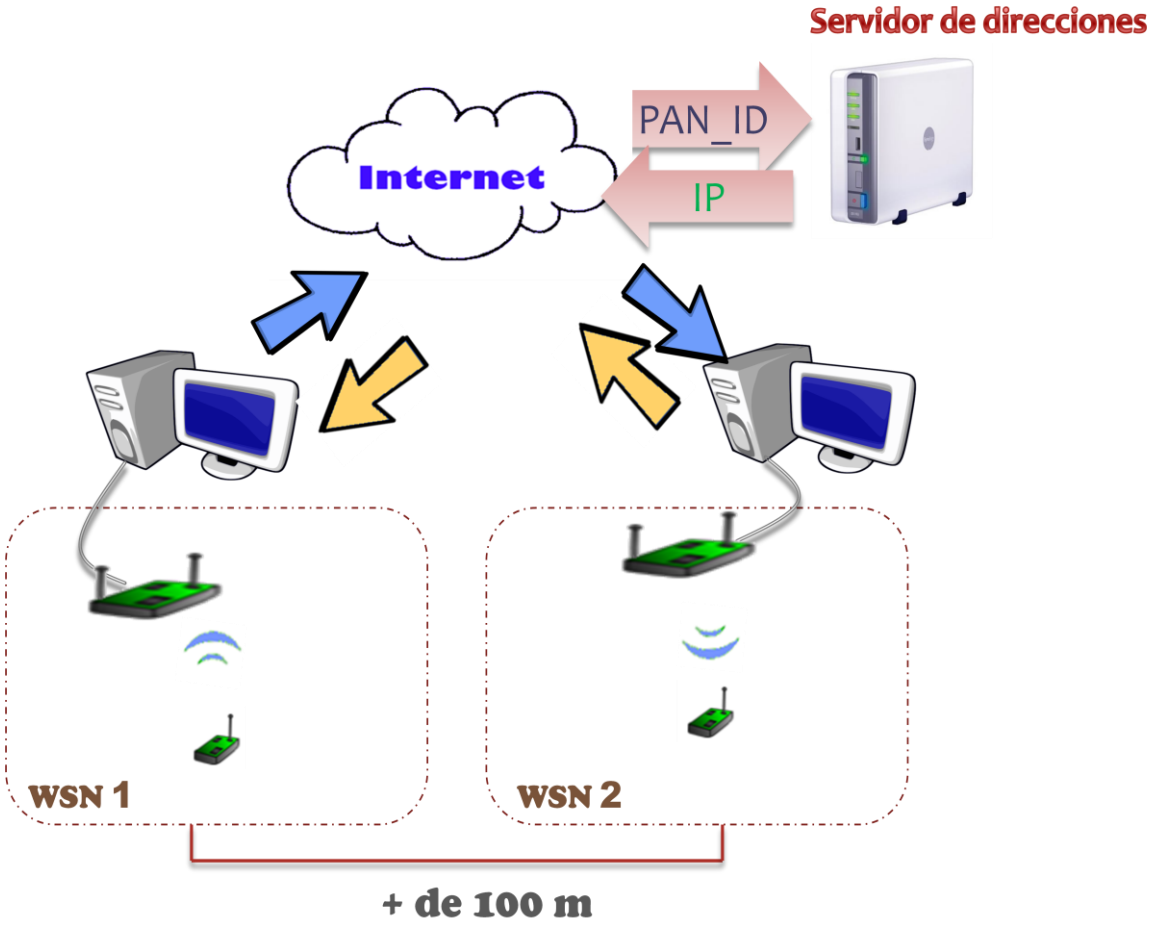


Ilustración 22: Diagrama de comunicación utilizando el servidor de direcciones.

7 Despliegue del escenario implementado

7.1 Introducción.

El escenario que se va a presentar a continuación constituye un ejemplo práctico de la aplicación de nuestra propuesta de integración de redes.

La funcionalidad de la aplicación será la más básica posible, simplemente se cambiara el estado de los led de los dispositivos con la recepción de la trama, esto nos permitirá ver claramente el intercambio de paquetes.

Tanto los Endpoint como el coordinador disponen de dos LED de estado y dos pulsadores que nos permiten interactuar fácilmente con el dispositivo. Para demostrar la comunicación simple entre dos dispositivos de la red, el pulsador 1 envía un mensaje a otro nodo conectado en su misma subred,(mismo PAN_ID) que supondrá el toggle del led 1 del dispositivo receptor del mensaje.

Ilustración 23: Comunicación local utilizando el pulsador 1.

Por el contrario el pulsador 2 genera un mensaje con destino a una subred remota, este será interpretado por el coordinador que encaminará el mensaje hacia el exterior. Para el dispositivo receptor el mensaje es idéntico al de un dispositivo conectado en su propia subred, por lo cual realizará un toggle en el LED al recibir el datagrama.

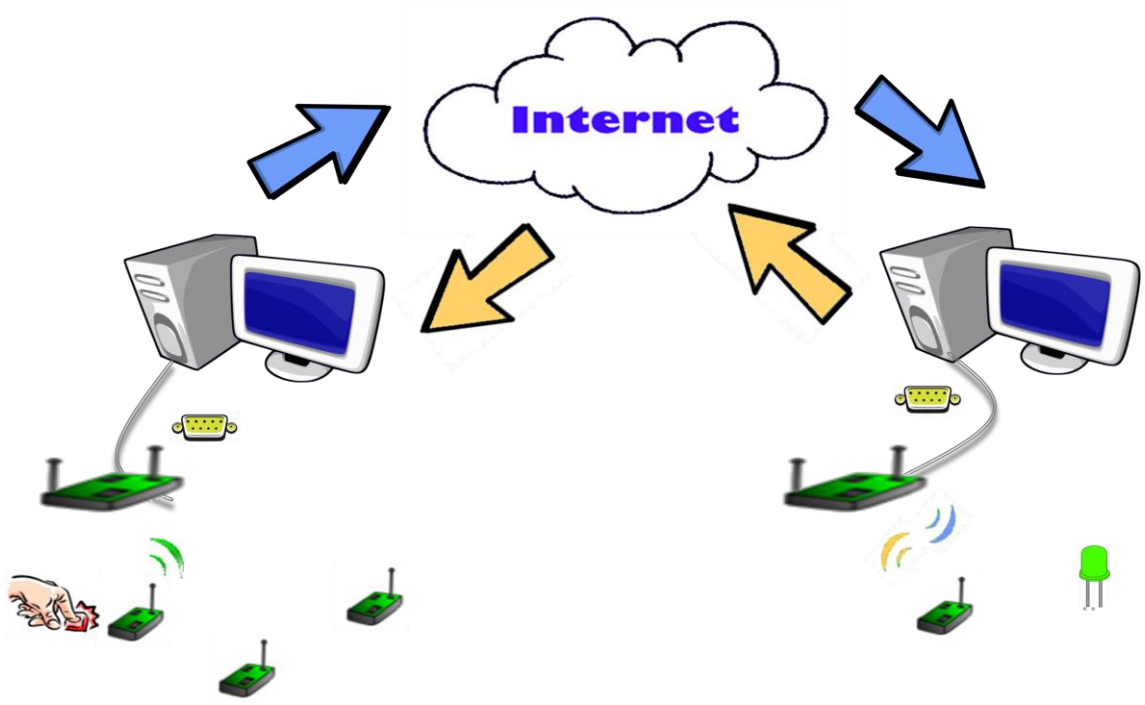


Ilustración 24: Escenario implementado

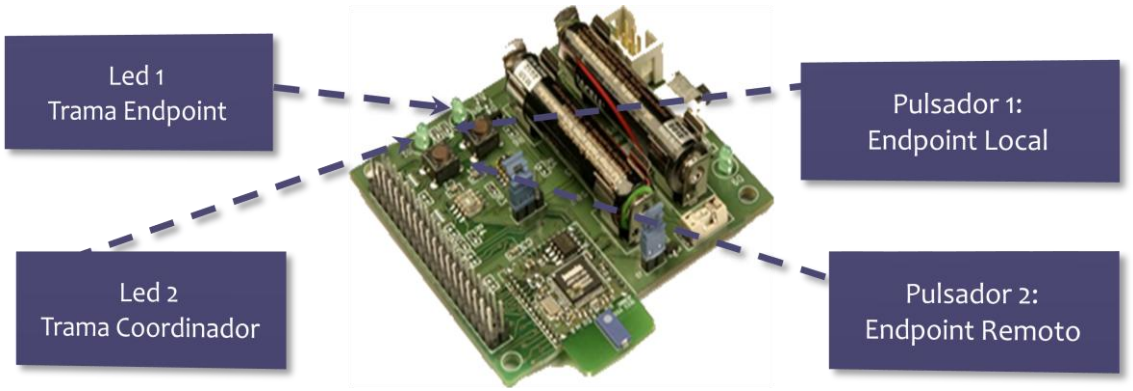


Ilustración 25: Detalle de la localización de pulsadores y Led en los EndPoint

7.2 Presentación del Hardware.

Para poder exponer el problema planteado en este proyecto debemos contar con un conjunto de dispositivos que soporten la solución planteada, así se utilizo el Kit de evaluación Jennic JN5139, que utiliza la tecnología 802.15.4, este principalmente incluye:

- 1 placa de control (coordinador).
  - Microprocesador y modulo de transmisión.
  - Dispone de alimentación mediante baterías o bien mediante conexión directa a la red.
  - Sensor de luz, temperatura y humedad.
  - Pantalla LCD de 128x64.
  - Conexión mediante puerto de serie.
  - 4 pulsadores y 4 led.
  - 1 pulsador de reseteo.

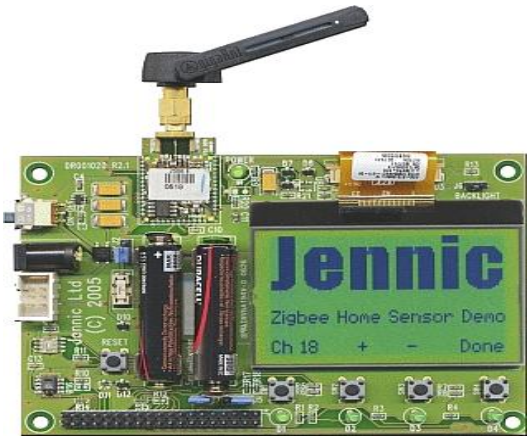


Ilustración 26: Placa con LCD y 4 pulsadores

- 4 placas de sensores (Motas).
  - Microprocesador y modulo de transmisión.
  - Dispone de alimentación mediante baterías.
  - Sensor de luz, temperatura y humedad.
  - Conexión mediante puerto de serie.
  - 2 pulsadores y 2 led.

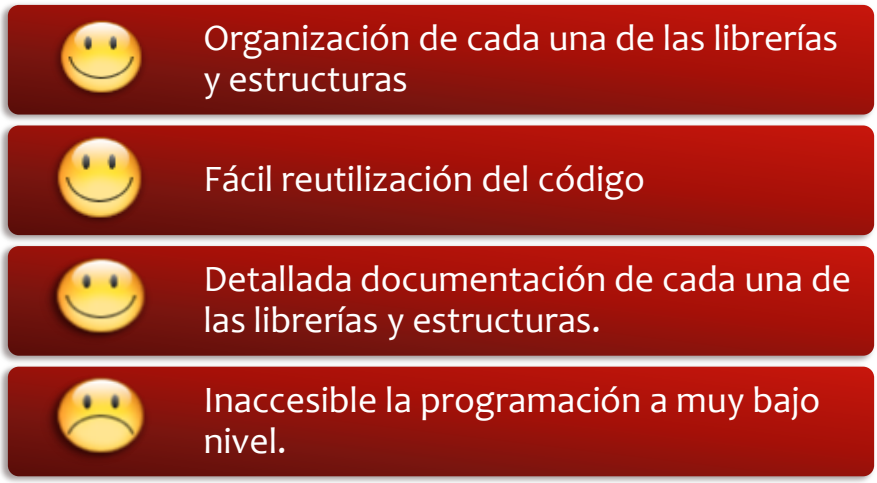
Ilustración 27: Placa de sensores

Más información:

http://www.jennic.com/products/development_kits/jn5139_ieee802154_jennet_evaluation_kit

Cada uno de los dispositivos de Jennic (tanto motas como coordinador) poseen un micro controlador que interpreta código C, de esta forma podemos desarrollar aplicaciones para estos módulos basándonos en las librerías proporcionadas por Jennic, estas librerías son responsables tanto del intercambio de mensajes a bajo nivel entre dispositivos como de la interacción con periféricos conectados al procesador como pueden ser leds, interruptores, sensores…

Ventajas y desventajas principales del organigrama de Jennic:



7.3 Módulos y clases principales.

El software que hemos desarrollado nos permite una conexión directa extremo a extremo. En primer lugar mencionar que una de las características del sistema es su simetría, cuando hablamos de simetría nos referimos al paralelismo entre los extremos, de forma que el software Java en los bridges (típicamente PC´s) es el mismo, así también las dos redes 802.15.4 interconectadas son también idénticas, permitiendo una completa comunicación bidireccional.

La siguiente figura muestra esta simetría.

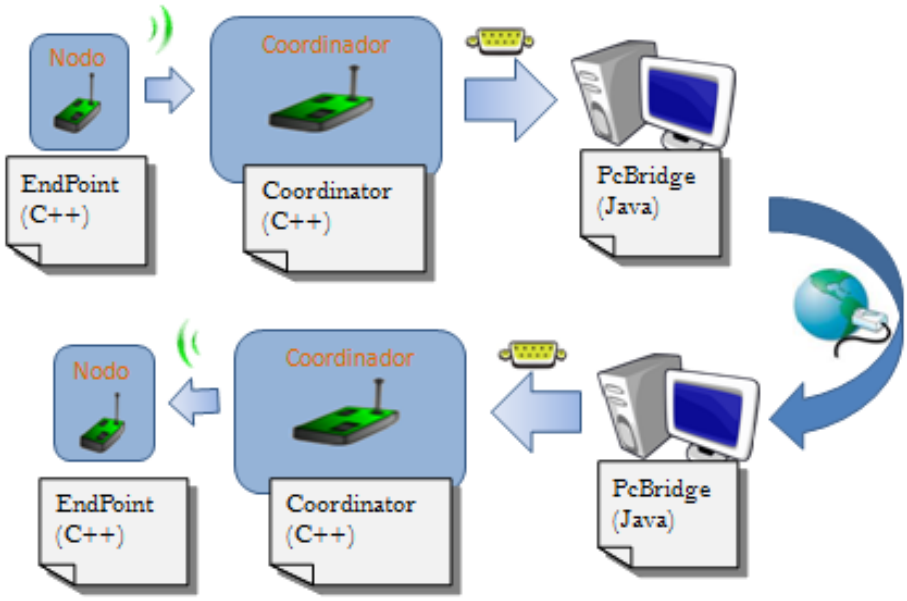


Ilustración 28: Situación del código desarrollado

La ilustración anterior muestra un esquema de los programas que será necesario desarrollar. Hablaremos primero del código C++ desplegado en los sensores, diferenciando entre Endpoint y Coordinador, en la segunda sección se describirá el funcionamiento de la clase Java que intercambia los paquetes a través de TCP/IP, además del servidor de direcciones.

7.3.1 Endpoint.

Como ya se ha comentado anteriormente, cuando hablamos de Endpoint nos referimos a lo que en el protocolo 802.15.4 se denomina como RFD. Estos dispositivos se comunican únicamente con el coordinador, el cual es responsable de direccionar los paquetes, de esta forma los dispositivos finales no tienen ninguna responsabilidad en cuanto a direccionamiento.

Para describir el comportamiento de los Endpoint podemos referirnos a dos fases, fase de inicialización y fase de acción.

La fase de inicialización tiene una duración determinada, comienza con el accionamiento del interruptor de encendido, y termina en el momento que el dispositivo es asociado y registrado en el coordinador.

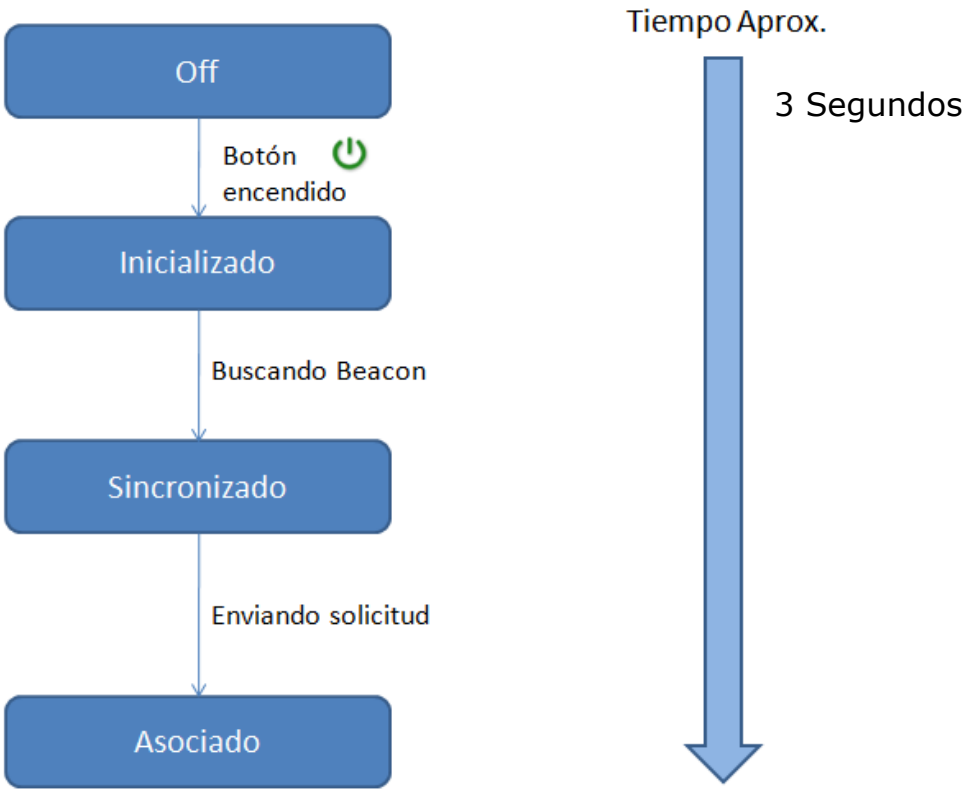


Diagrama 6: Diagrama del comportamiento del Endpoint en fase inicialización.

Esta secuencia se repite cada vez que el dispositivo pierde conexión con el coordinador.

La asignación de direcciones para cada una de las motas es responsabilidad únicamente del coordinador, que mantiene una tabla con las direcciones de los dispositivos sincronizados y sus direcciones MAC, asignando direcciones a las solicitudes secuencialmente desde una dirección base o inicial de asignación. Normalmente los coordinadores tiene un número máximo de dispositivos asociados, de esta forma se garantiza el funcionamiento óptimo de la red.

El PAN_ID es configurado en la memoria flash de cada una de las motas, los dispositivos en el inicio esperan recibir una trama Beacon desde el coordinador responsable de la PAN_ID que les permita sincronizarse. En cualquier caso esto podría establecerse como configurable, esto permitiria que las motas pudiesen conectarse a una u otra subred dependiendo de la señal recibida, o bien por cualquier otro motivo. En este proyecto hemos descartado esta opción y las motas tienen una PAN_ID establecida de inicio, esto nos permite una mayor robustez al asegurarnos que los dispositivos se conectaran únicamente a la red programada.

La fase de Acción no tiene una duración determinada, pues es el ciclo donde el dispositivo desarrolla su función. Generalizando podríamos hablar de dos modos de acción diferentes.

El primero se conoce como **modo "polling"**, donde el módulo despierta en intervalos regulares para normalmente realizar una lectura de sus mediciones o alguna comprobación, posteriormente transmite la información (si procede) y volver a un estado de letargo.

El segundo modo será el utilizado en el ejemplo de este proyecto, es conocido como **modo "evento"**, donde el modulo despierta por medio de una interrupción, la cual puede ser lanzada desde alguno de los periféricos del dispositivo (botones, sensores, timers...), el siguiente diagrama muestra el comportamiento para el dispositivo de nuestro ejemplo:

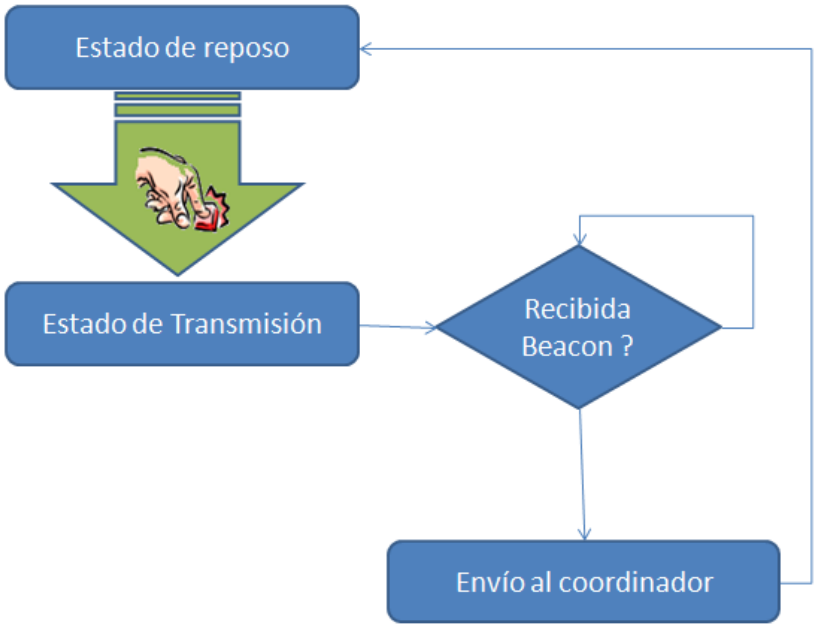


Diagrama 7: Diagrama del comportamiento del Endpoint en fase acción.

7.3.1.1 Detalle del código C en el Endpoint (ButtonDemoEndpt.c)

La fase de inicialización de los Endpoint se realiza en los métodos:

```
PRIVATE void   vInitSystem(void);

PRIVATE void   vInitEndpoint(void);

PRIVATE void   vStartScan(void);

PRIVATE void   vStartSync(void);
```

Estos son invocados por el método *AppColdStart* (), que es el punto de partida en la ejecución del código.

No entraremos demasiado en detalle en la fase de inicialización de los dispositivos pues no es el propósito del proyecto, simplemente

mencionar que en esta fase se inician los procesos que nos permitirán utilizar periféricos como el puerto serie (UART), Leds, etc. Además de establecer la sincronización con el coordinador.

Durante la fase de inicialización los dos Leds de la placa permanecerán encendidos, estos cambiaran su estado (apagados) cuando el proceso de inicialización haya terminado. Si el Endpoint perdiese la sincronización con el coordinador, volvería a la fase de sincronización encendiendo los dos leds simultáneamente de nuevo.

En la fase que hemos denominado como de acción es donde se define realmente la tarea de nuestro Endpoint, la cual en nuestro ejemplo será simplemente capturar el accionamiento de uno de los pulsadores, y enviar una trama a un nodo de su red o bien a otro en una red diferente.

Cuando una de los dos pulsadores es accionado se presenta una interrupción, esto desencadena una serie de acciones que permiten enviar la trama al destino.

Como se ha mencionado, una de las particularidades más destacadas de 802.15.4 es la eficiencia energética. Para ello las motas permanecen en estado de reposo la mayor parte del tiempo, despertando en los tiempos de Beacon. Siguiendo este comportamiento, el accionamiento de uno de los pulsadores será capturado por el método:

```
PRIVATE bool_t bProcessKeyPress(void)
```

Este determina el botón pulsado, y simplemente cambia el estado de una variable.

Ejemplo de pulsación en el botón 0:

```
sDemoData.sSystem.bSw0State = TRUE;
```

Esta variable será procesada cuando se reciba la Beacon del coordinador, es este momento cuando el Endpoint está "despierto" y realiza sus tareas, en nuestro caso el dispositivo comprueba si se ha presionado alguno de los pulsadores.

El método activado en la recepción de las tramas de Beacon esta preestablecido en las librerías de Jennic y se denomina:

```
PRIVATE void vProcessRxBeacon(MAC_MlmeDcfmInd_s *psMlmeInd)
```

En nuestro ejemplo es realmente sencillo. Comprobaremos si alguno de los dos pulsadores ha sido presionado, en el caso de haberse presionado el pulsador 1 se envía un datagrama a un Endpoint fuera de la subred, el segundo botón envía el datagrama a la otra mota de la misma WSN.

```
if (sDemoData.sSystem.bSw0State == TRUE)

{

    /* Enviamos la trama a un EndPoint fuera de nuestra red*/


    vSendMessage(ENDPOINT1 ,PAN_ID_REMOTE, ( 100 + 2 + 8 ), aux);
}

else if (sDemoData.sSystem.bSw1State == TRUE)

{

    /* Enviamos la trama a la otra mota de la WSN*/

    if (sDemoData.sSystem.u8ThisNode == ENDPOINT1)

    {

        vSendMessage(ENDPOINT2 ,PAN_ID, ( 100 + 2 + 8 ), aux);
    }

    else

    {

        vSendMessage(ENDPOINT1 ,PAN_ID, ( 100 + 2 + 8 ), aux);
    }

}
```

El datagrama enviado, además de contener la información correspondiente para su enrutamiento, contiene 100 bytes de información irrelevante, el único propósito de esto es incluir datos en el datagrama.

El método "sendMessage" recibe como argumentos:

- ShortAddress de la mota destino.
- Identificador de red de destino (PAN_ID).
- Tamaño del datagrama.
- Datos a enviar

```
PRIVATE void vSendMessage(uint8 u8Node,uint16 dst_PanId ,uint8 sduLength, char *info)
```

La cabecera UC3M está compuesta por 8 bytes, añadidos dentro del "payload" de 802.15.4, el archivo *Uc3m_Proto.h* contiene la estructura del protocolo.

Uc3m_Proto.h

```
typedef struct uc3m_header {

    uint16 u16source_Addr;
    uint16 u16destinationAddr;
    uint16 u16source_panid;
    uint16 u16destination_panid;
    uint8 data [102];

}uc3m_header_t;
```

Las tramas 802.15.4 transmiten como máximo 110 bytes, podemos ver como se han reservado 102 bytes, pues los 8 restantes contienen la información del protocolo Uc3m.

Una vez establecidas las cabeceras de 802.15.4 podemos incluir la cabecera Uc3m.

```
//Creamos nuestra cabecera UC3M
struct ping_msg ping_app ;
ping_app.type = PING_TYPE;

memcpy((char *)info,ping_app.data, 102);
ping_app.length = sduLength - 8;

struct uc3m_header uc3m_msg ;
//Rellenamos la cabecera uc3m
uc3m_msg.u16source_Addr = (sDemoData.sSystem.u16ShortAddr);
uc3m_msg.u16destinationAddr = htons(u16DestAddr);
uc3m_msg.u16source_panid = htons(PAN_ID);
//Le decimos el panID destino
    uint16 pan = dst_PanId;
uc3m_msg.u16destination_panid = htons(pan);

memcpy((char *)&ping_app , uc3m_msg.data, ping_app.length);
memcpy((char *)&uc3m_msg.data,pu8Payload, 110);


pu8Payload[0] = (uint8)  sDemoData.sSystem.u16ShortAddr ;
pu8Payload[1] = (uint8) (sDemoData.sSystem.u16ShortAddr >> 8 );

pu8Payload[2] = (uint8)  u16DestAddr ;
pu8Payload[3] = (uint8) (u16DestAddr >> 8 );
//Informamos del pan_ID destino
uint16 panid = dst_PanId;
pu8Payload[4] = (uint8)  panid ;
pu8Payload[5] = (uint8) (panid >> 8 );
```

Para concluir la llamada a los métodos que coordinan el envío y recepción de mensajes entre los dispositivos.

```
vAppApiMcpsRequest(&sMcpsReqRsp, &sMcpsSyncCfm);
sDemoData.sSystem.eState = E_STATE_TX_DATA;
```

Los Endpoint también están capacitados para recibir tramas de datos, originando un cambio de estado en uno de sus leds, dependiendo si la trama tiene como origen el coordinador o por el contrario procede de otro Endpoint.

Las librerías de Jennic nos ofrecen un método el cual será activado con la recepción de una trama.

```
PRIVATE void vProcessIncomingMcps(MAC_McpsDcfmInd_s *psMcpsInd)
```

En primer lugar hay que diferenciar entre tramas de confirmación (ACK), y tramas de datos.

Para las tramas de ACK simplemente enviaremos un mensaje por el puerto serie para la monitorización.

```
if ((psMcpsInd->u8Type == MAC_MCPS_DCFM_DATA)
    && (sDemoData.sSystem.eState == E_STATE_TX_DATA))
{
        vPrintString("recibido un ACK ");
```

Nota: internamente se realizaran otro tipo de operaciones como el incremento en el número de secuencia de la trama.

Las tramas de datos supondrán un cambio de estado en el Led correspondiente, para determinar la procedencia procesamos el datagrama.

7.3.2 Coordinador

La función del coordinador es esencial para la comunicación de los sensores, tanto dentro de la subred, como fuera de esta.

El enrutamiento de paquetes de la red, es responsabilidad del coordinador, en este caso las motas están configuradas para enviar todos los datagramas al coordinador, y este es quien los encamina.

Además, es responsable del proceso de ”beaconing”, esto significa que es tarea del coordinador enviar periódicamente las tramas de Beacon que gobiernan la comunicación en las redes 802.15.4, estas tramas como ya se ha descrito anteriormente permiten la tanto la sincronización de nuevos nodos, como el intercambio ordenado de datagramas, además permiten al coordinador mantener el estatus de los nodos de su subred.

El siguiente diagrama muestra el procedimiento del coordinador:

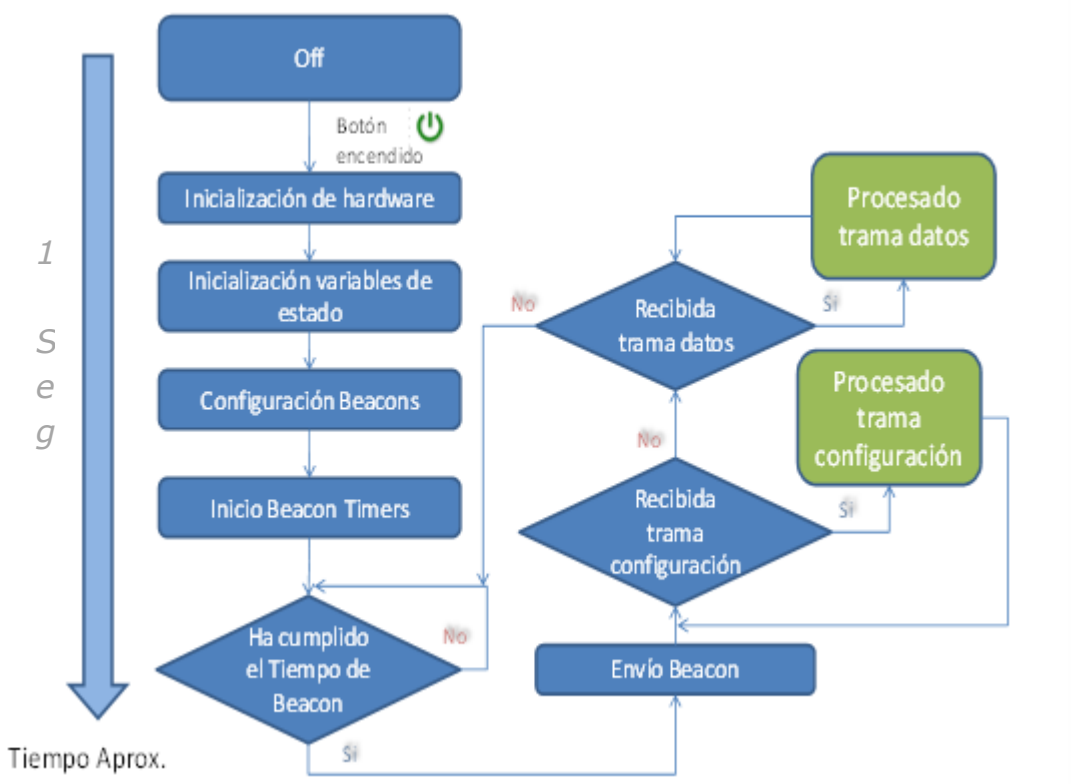


Diagrama 8: Diagrama del funcionamiento del coordinador.

Conjuntamente, el coordinador será el responsable de interpretar la información de direccionamiento UC3M para decidir la ruta del paquete.

El siguiente diagrama muestra el procesado de la trama de datos por parte del coordinador.

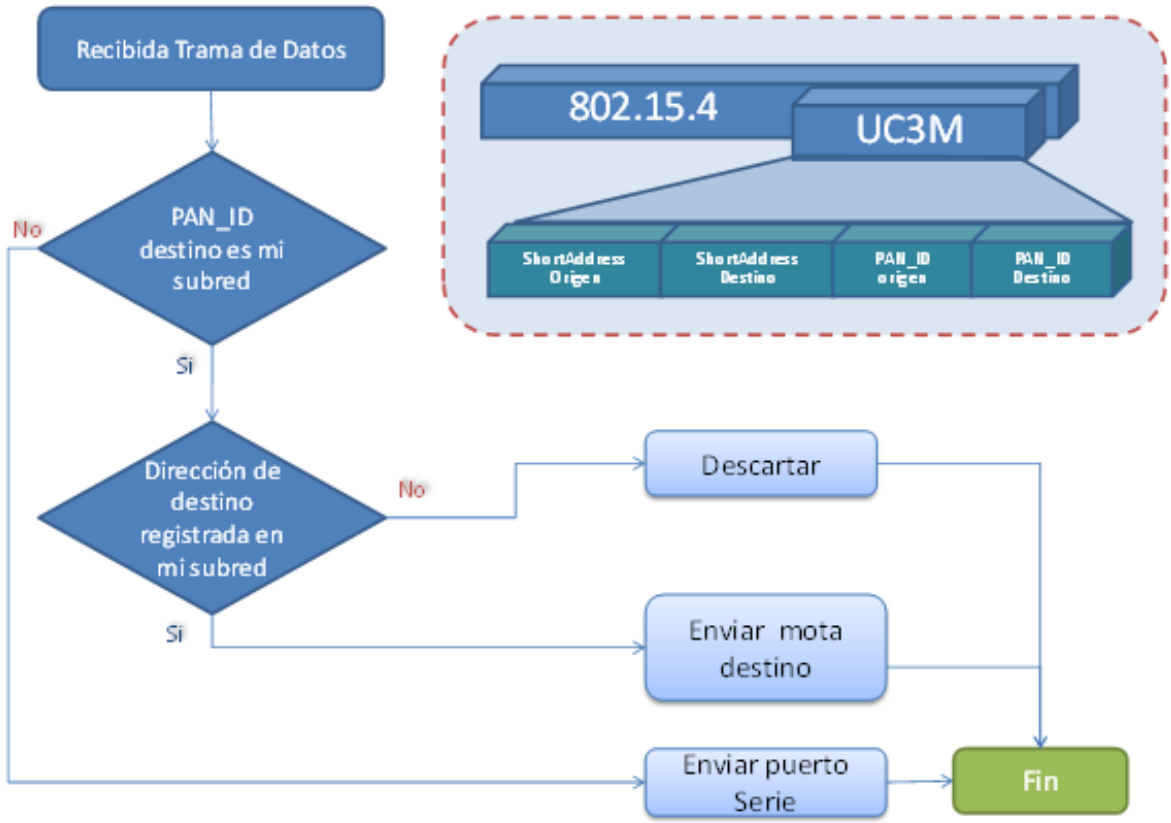


Diagrama 9: encaminamiento de las tramas por parte del coordinador.

Siguiendo la secuencia del diagrama, podemos ver claramente como a la entrada de una trama de datos, el coordinador analiza secuencialmente las cabeceras para tomar una decisión. Existen tres opciones posibles.

- **Descartar**: La trama se pierde.
- **Enviar a destino:** En este caso el destinatario está directamente asociado al coordinador
- **Enviar por el puerto de serie**: El destinatario se encuentra en una localización remota, por lo que se reenvía al PC.

7.3.2.1 Detalle del código C en el Coordinador (ButtonDemoCoord.c)

Como es lógico, el código es mucho más complejo que en los Endpoint, utilizaremos un fichero de configuración donde definiremos las constantes que servirán para configurar el coordinador, este fichero tendrá el nombre de "*ButtonDemoConfig.h*". A continuación se presentan algunas de las constantes definidas:

NODE_ID será el primer byte en el Payload de las tramas enviadas por los nodos, esto servirá al coordinador para identificar las tramas de los nodos.

```
#define NODE_ID                                 0x5b
```

PAN_ID, identificador de la red 802.15.4 que despliega el coordinador.

```
#define PAN_ID                                  0xde35
```

Número máximo de Endpoint que pueden asociarse.

```
#define MAX_ENDPOINTS                           2
```

Dirección corta del coordinador.

```
#define COORD_SHT_ADDR                          0x0e00
```

Dirección base de los Endpoint, esta será la primera dirección asignada por el coordinador, después irá incrementando esta base uno a uno.

```
#define ENDPOINT_SHT_ADDR_BASE                  0xf000
```

Primeros 4 bytes de la dirección larga del coordinador.

```
#define COORD_EXT_ADDR_LO                       0x00000000
```

Últimos 4 bytes de la dirección larga del coordinador.

```
#define COORD_EXT_ADDR_HI                    0x00000f1c
```

Establece el canal de radio.

```
#define DEFAULT_CHANNEL                      20
```

El resto de definiciones están más relacionadas con conceptos de variables estáticas de programación.

A continuación detallaremos las dos fases de la vida del coordinador tal y como describimos anteriormente.

La fase de inicialización en el coordinador se realiza en el método:

```
PRIVATE void vInitSystem(void)
```

Dentro de este método se realiza tanto la inicialización de los componentes de hardware del dispositivo (funciones internas de las librerías de Jennic), hasta la inicialización del proceso de "Beaconing". Algunos comentarios acerca de la inicialización:

Esta sentencia nos permite interrumpir los procesos del coordinador a través del puerto serie, esto es de vital importancia cuando necesitamos comunicar a un coordinador que ha recibido una trama de otra subred con destino a una de sus motas.

```
vAHI_Uart0RegisterCallback(vRxDatos);
```

En la inicialización de la subred es necesario definir el identificador de la red (PAN_ID), además utilizamos direcciones cortas (2 bytes).

```
MAC_vPibSetPanId(s_pvMac, PAN_ID);

MAC_vPibSetShortAddr(s_pvMac, COORD_SHT_ADDR);
```

Asignación del canal de la red, por defecto asignaremos el canal 20.

```
sCoordData.sSystem.u8Channel = DEFAULT_CHANNEL;
```

Por último el coordinador debe iniciar el proceso de "Beaconing", las librerías internas de Jennic realizan este proceso por nosotros, únicamente podemos configurar algunos parámetros.

Con esto se daría por finalizada la fase de inicialización, a partir de ahora el coordinador está preparado para sincronizar nuevos nodos y gestionar y encaminar los paquetes de estos.

Para la sincronización de un Endpoint, es necesario que este envíe un "MLME Associate request", es decir, una solicitud de ingreso en la red. Esto desata un evento en el coordinador:

```
PRIVATE void vProcessIncomingMlme(MAC_MlmeDcfmInd_s *psMlmeInd)
```

En nuestro ejemplo, si no se ha superado el número máximo de dispositivos asociados el coordinador responde con una trama de confirmación:

```
vHandleNodeAssociation(psMlmeInd);
```

A partir de la sincronización, el Endpoint está capacitado para enviar datos al coordinador, estos son procesados por el coordinador mediante el método:

```
PRIVATE void vProcessIncomingData(MAC_McpsDcfmInd_s *psMcpsInd)
```

Hay dos tipos de tramas que pueden ser capturadas por este método, las tramas de confirmación (ACK) y las tramas de datos, por lo tanto, debemos separarlas convenientemente, para ello podemos utilizar "Type" de la trama recibida.

Las tramas de confirmación (ACK) son tratadas por las librerías propietarias de Jennic, por lo que no tenemos acceso a la configuración. Tanto en los coordinadores como en los Endpoint, serán utilizadas principalmente para incrementar números de

secuencia en el intercambio de mensajes. Para esta demostración se enviará por el puerto serie un mensaje avisando de que se ha recibido una trama de confirmación, esto nos permitirá seguir el intercambio de mensajes desde la consola.

```
if (psMcpsInd->u8Type == MAC_MCPS_DCFM_DATA)
{
vPrintString("soy un ACK al coord origen  ");

vPrintString("\r\n");
```

Las tramas de datos recibidas por el coordinador constituyen una parte muy importante del proyecto, ya que deben ser procesadas para poder ser reenviadas.

Antes de resolver el direccionamiento del paquete se interpretan la cabecera Uc3m del paquete y se envía la información por el puerto de serie para poder ser monitorizada.

```
//Tratamos la cabecera Uc3m
uint8 proto15_4Payload [102];
struct uc3m_header * ptr;
ptr = (struct uc3m_header *) proto15_4Payload;
ptr = htons(ptr);
u16ScrRealAddr = (ptr->u16source_Addr);
u16DstRealAddr = htons(ptr->u16destinationAddr);
u16Dst_panid = htons(ptr->u16destination_panid);
//-----------------

u16ScrRealAddr=(psFrame->au8Sdu[1] << 8) + psFrame->au8Sdu[0];
u16DstRealAddr=(psFrame->au8Sdu[3] << 8) + psFrame->au8Sdu[2];
u16Dst_panid=(psFrame->au8Sdu[5] << 8) + psFrame->au8Sdu[4];

            vPrintString("-----------------------------------------------------\r\n");
            vPrintString("-------802.15.4-------\r\n");
            vPrintString("Source: ");
            vNumToString(u16NodeAddr, au8TmpBuffer);
            vPrintString(au8TmpBuffer);
            vPrintString(" Destiny: ");
            vNumToString(u16NodeAddrDst, au8TmpBuffer);
            vPrintString(au8TmpBuffer);

            vPrintString("\r\n");
            vPrintString("---------UC3M---------\r\n");
            vPrintString("Origen: ");
            vNumToString(u16ScrRealAddr, au8TmpBuffer);
            vPrintString(au8TmpBuffer);

            vPrintString(" Destino: ");
            vNumToString(u16DstRealAddr, au8TmpBuffer);
            vPrintString(au8TmpBuffer);


            vPrintString("  PAN_ID_dest: ");
            vNumToString( u16Dst_panid, au8TmpBuffer);
            vPrintString(au8TmpBuffer);
            vPrintString("\r\n");
            vPrintString("-----------------------------------------------------\r\n");
```

Posteriormente, como ya se ha puntualizado existen tres posibles casos, a continuación se describirá como se resolverá cada uno y la parte del código correspondiente.

Caso 1: El identificador PAN_ID de la cabecera uc3m no corresponde con el PAN_ID de mi subred, por lo tanto el paquete debe ser reenviado al PC por el puerto serie, y será este quien decida sobre su encaminamiento.

Para poder enviar el datagrama por el puerto serie es necesario diferenciar entre los mensajes de monitorización que informan sobre el estado de las transmisiones y los datos enviados para su redirección, para esto se ha creado un sencillo protocolo con carácter de inicio (#) y fin ($), que delimita la secuencia de caracteres con la información del datagrama.

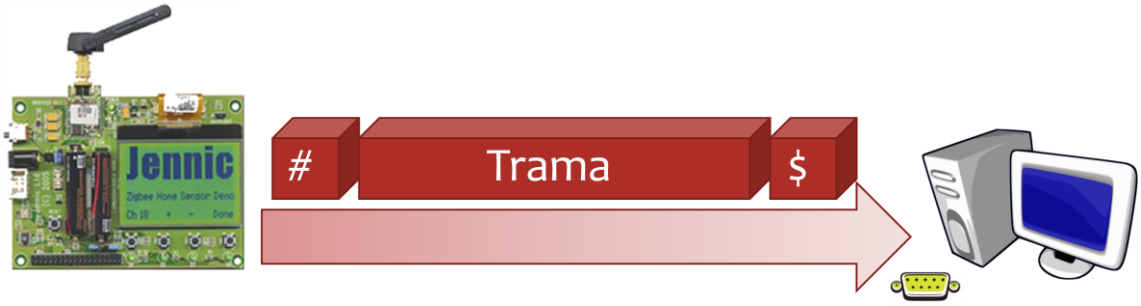


Ilustración 29

```
}else if(!myPan){

    vPrintString(" No es para mi  PAN_ID !!!! \r\n");
    vPrintString(" Envio por El puerto serie.... \r\n");

    vPrintString("#");
    //envio short Origen, destino
    vNumToString(u16ScrRealAddr, au8TmpBuffer);
    vPrintString(au8TmpBuffer);
    vNumToString(u16DstRealAddr, au8TmpBuffer);
    vPrintString(au8TmpBuffer);
    //OPAN
    vNumToString(PAN_ID, au8TmpBuffer);
    vPrintString(au8TmpBuffer);
    //DPAN
    uint16 pAN_D = (psFrame->au8Sdu[5] << 8) + psFrame->au8Sdu[4];
    vNumToString(pAN_D, au8TmpBuffer);
    vPrintString(au8TmpBuffer);
    //Envio tamaño de los datos
    vNumToString(0x0002, au8TmpBuffer);
    vPrintString(au8TmpBuffer);

    vPrintString("$");


}
```

Caso 2: El identificador PAN_ID de la cabecera uc3m corresponde con el PAN_ID de mi subred, y además la dirección de destino es la de el coordinador, esto significa que el datagrama es para el propio coordinador, por lo que no necesita ser encaminado, simplemente será enviado un mensaje por el puerto serie para la monitorización.

```
if(u16DstRealAddr == COORD_SHT_ADDR && u16Dst_panid == PAN_ID ){
    vPrintString("El paquete es para mi que soy el coord de la red");
```

Caso 3: El identificador PAN_ID de la cabecera uc3m corresponde con el PAN_ID de mi subred, y en este caso la dirección de destino no es la dirección del coordinador, esto significa que la comunicación es entre dos nodos de la WSN del coordinador, por lo que el paquete debe ser reenviado por RF a su destinatario final.

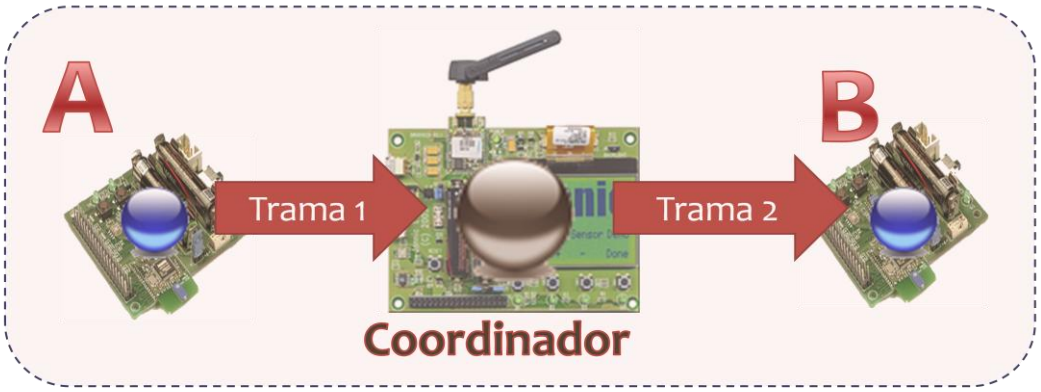
```
if(mySon && myPan){
vPrintString("Reenvio el paquete a una de mis motas");
```

El coordinador reenvía la trama prácticamente igual que la ha recibido, la cabecera uc3m se mantiene intacta, de esta forma el dispositivo que recibe el paquete conoce el emisor real de la trama.

```
pu8Payload[0] = psFrame->au8Sdu[0];
pu8Payload[1] = psFrame->au8Sdu[1];

pu8Payload[2] = psFrame->au8Sdu[2];
pu8Payload[3] = psFrame->au8Sdu[3];
```

Únicamente se modifican los campos de las cabeceras origen y destino 802.15.4, que permiten el direccionamiento dentro de la WSN, en este caso el origen será la dirección del coordinador y el destino la dirección que figure en el destino del campo uc3m.



	Origen 802.15.4	Destino 802.14.4	Origen Uc3m	Destino Uc3m
Trama 1	A	Coordinador	A	B
Trama 2	Coordinador	B	A	B

Ilustración 30

```
/* direccion de origen */

sMcpsReqRsp.uParam.sReqData.sFrame.sSrcAddr.u8AddrMode = psFrame->sSrcAddr.u8AddrMode;
sMcpsReqRsp.uParam.sReqData.sFrame.sSrcAddr.u16PanId = psFrame->sSrcAddr.u16PanId;
sMcpsReqRsp.uParam.sReqData.sFrame.sSrcAddr.uAddr.u16Short = COORD_SHT_ADDR  ;

/* Direccion de destino */

sMcpsReqRsp.uParam.sReqData.sFrame.sDstAddr.u8AddrMode =psFrame->sDstAddr.u8AddrMode;
sMcpsReqRsp.uParam.sReqData.sFrame.sDstAddr.u16PanId = psFrame->sDstAddr.u16PanId;
sMcpsReqRsp.uParam.sReqData.sFrame.sDstAddr.uAddr.u16Short = u16DstRealAddr;
```

7.3.3 PC como punto de acceso 802.15.4

Como ya se ha comentado en este documento, es necesario disponer de al menos dos maquinas conectadas a la red para tramitar el tránsito de datos entre las dos redes WSN, sobre estas máquinas se ejecutará el programa Java PC Bridge.

La razón por la que se ha escogido desarrollar esta aplicación en Java y no en cualquier otro lenguaje es simplemente el hecho de la facilidad de uso de sus librerías y su compatibilidad con el paquete RxTx, el cual nos permite la interpretación de caracteres desde el puerto serie.

Para simplificar y optimizar el funcionamiento de la aplicación se han desarrollado dos hilos Java que se reparten las tareas, además la resolución de direcciones queda delegada a un servidor de direcciones de forma remota e independiente.

De esta forma la aplicación PCBridge se inicia tomando como único parámetro de configuración el puerto serie donde estará conectado el coordinador(Ejemplo COM1). Una vez establecida la conexión con el coordinador se despliegan dos hilos Java independientes, asignado el flujo de entrada a "SerialReader" y el flujo de salida a "SerialWriter".

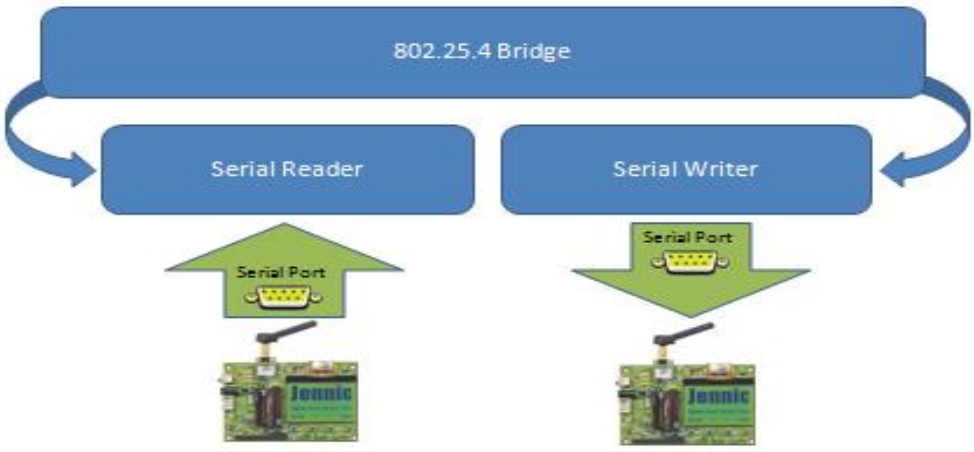


Diagrama 10: Esquema de la comunicación por puerto de serie.

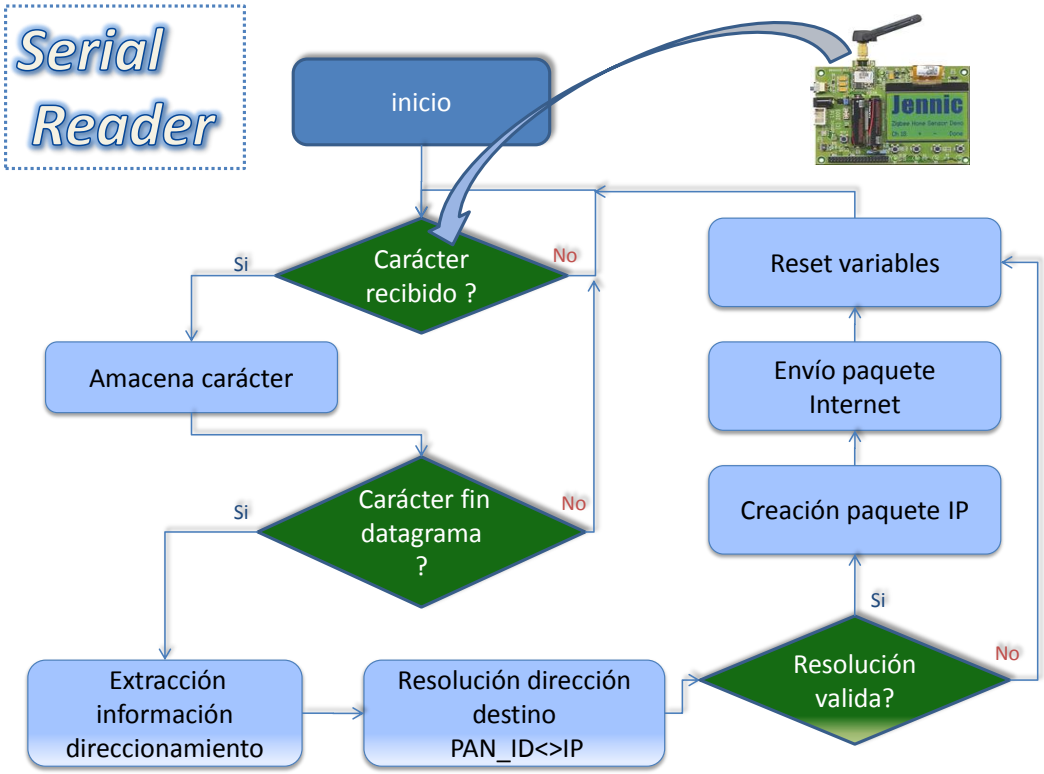


Diagrama 11: Diagrama de funcionamiento del hilo Java SerialReader.

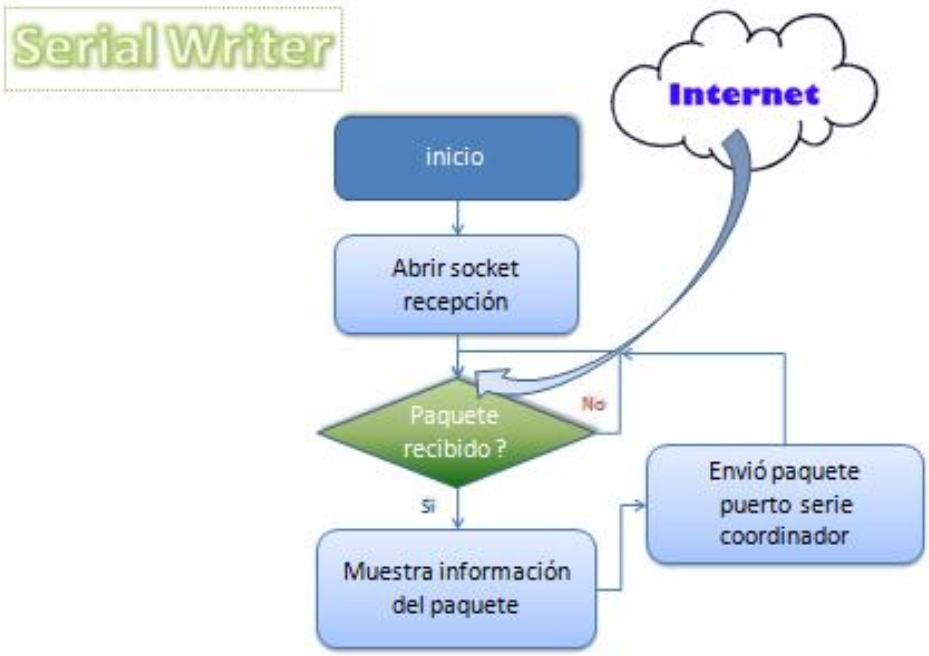


Diagrama 12: Diagrama de funcionamiento del hilo Java SerialWriter.

Para tener constancia del comportamiento del sistema se introducen sentencias en el coordinador que envían por el puerto serie cadenas de caracteres. La aplicación descartara estas cadenas ya que no se ajustan a las secuencias establecidas por el protocolo como tramas, y las mostrara por la consola del PC.

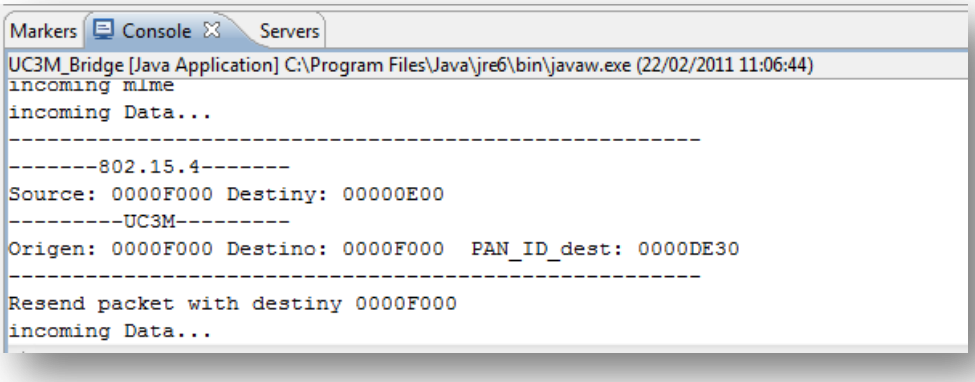


Ilustración 31: Ejemplo de monitorización del sistema.

7.3.3.1 Código PC Bridge

En el apartado anterior ya se ha hablado acerca de PC Bridge, la aplicación Java responsable de enlazar la red TCP/IP, con las WSN, a través del puerto de serie. Ahora en esta sección se puntualizara acerca de algunos detalles destacables del código.

PC Bridge está dividido en cuatro clases principales que conforman toda la funcionalidad de la aplicación.

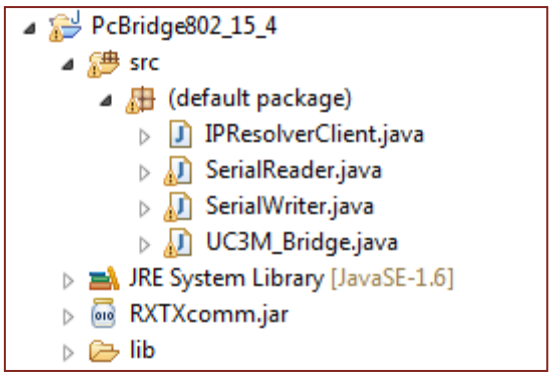


Ilustración 32: Perspectiva de eclipse del proyecto.

UC3M_Brige.java contiene el método "Main" de la aplicación, su cometido es únicamente reservar el puerto de comunicación serie, y lanzar los hilos de lectura y escritura, asignándoles los flujos de entrada y salida de datos.

A continuación podemos ver algunos fragmentos significativos de código de la clase.

Comprobación del puerto de serie COM

```
if ( portIdentifier.isCurrentlyOwned() )
{
   System.out.println("Error: Port is currently in use");
}
else
{
   CommPort commPort = portIdentifier.open(this.getClass().getName(),2000);
```

Apertura y reserva del Puerto COM.

```
if ( commPort instanceof SerialPort )
{
   SerialPort serialPort = (SerialPort) commPort;
   serialPort.setSerialPortParams(19200,SerialPort.DATABITS_8,SerialPort.STOPBITS_1,SerialPort.PARITY_NONE);
```

Inicio de hilos y asignación de flujos de datos.

```
   InputStream in = serialPort.getInputStream();
   OutputStream out = serialPort.getOutputStream();

   (new Thread(new SerialReader(in))).start();
   (new Thread(new SerialWriter(out))).start();
```

SerialReader.Java permanece a la espera de la entrada de datos por el puerto de serie.

```
while ( ( len = this.in.read(buffer)) > -1 )
```

Estos datos pueden ser de dos tipos. Por una parte podemos recibir **Mensajes de monitorización**, los cuales son publicados en la consola de la aplicación sin ser procesados, pues únicamente sirven para mostrar el funcionamiento de la WSN.

El otro tipo son **Tramas de datos**, las cuales han de ser procesadas y encaminadas, consultando la IP destino al servidor de direcciones. Las tramas de datos vienen precedidas del carácter de inicio #, seguidas de 40 caracteres, finalizando en $.

Por lo tanto, debemos comprobar si la información recibida desde el coordinador es realmente una trama.

```
if(new String(buffer,0,len).equals("#")){
   inicio_almohadilla = true;
}
```

Concatenamos los 40 caracteres de la trama que contendrán la información de envío.

```
if(inicio_almohadilla){
   cont++;

   pKt = pKt.concat(new String(buffer,0,len));
}
```

```
if(cont == 40){ //Ya me ha entrado la cabecera

    message=new String(pKt);
    msg=message.getBytes();
    destination = message.substring(24, 32);
    origin = message.substring(16, 24);
    longi = Integer.parseInt(message.substring(32, 40));
```

A partir del identificador de WSN destino, podemos realizar una consulta al servidor de direcciones para obtener la dirección IP de la maquina responsable de encaminar los paquetes para esa WSN.

Se ha implementado una clase independiente "IPResolverClient.java" la cual efectuara la resolución de direcciones con el servidor, esta clase será detallada más adelante junto con el servidor de direcciones. De esta forma la consulta se ejecuta con la llamada al método "getRemoteIP", facilitándole los Identificadores PAN_ID.

```
iPDestination=IPresolver.getRemoteIP(origin, destination);
System.out.println("Tengo destino !!!!!!!!"+ iPDestination);
```

En el caso de que el servidor de direcciones no contenga la IP solicitada nos enviara la dirección 0.0.0.0, esta será interpretada como dirección nula lo que mostrara por la consola.

```
System.out.print("No puedo enviar !!!!!!! ");
System.out.print("No se puede encontrar asociacion entre PAN_ID--IP");
System.out.println(destination);
```

Una vez obtengamos la dirección IP, podemos enviar el datagrama con la información estableciendo un socket al puerto 5025 de la maquina remota.

```
if(iPDestination != null  && iPDestination !="0.0.0.0" ){
   paquete=new DatagramPacket(msg,msg.length,InetAddress.getByName(iPDestination),PUERTO);
   System.out.print("Datagrama listo para enviar.....");
   s.send(paquete);
   System.out.println("Enviando a  "+InetAddress.getByName(iPDestination));
```

Desde el otro extremo de la comunicación, SerialWriter.Java espera los paquetes entrantes en el puerto 5025. Cada vez que recibe un datagrama, muestra por pantalla los datos recogidos, y posteriormente lo reenvía carácter a carácter por el puerto seria hacia su coordinador.

```
recibido = new DatagramPacket(new byte [1024],1024);
//datagram comes
try{
      s.receive(recibido);
}catch(Exception e){
      System.out.println("Error en la recepcion");
}
System.out.println("Request is comming \n");
System.out.println("Origin :" + recibido.getAddress());
System.out.println("Port :" + recibido.getPort());
String received = new String (recibido.getData());
System.out.println ("data: " + received.substring(1, 40) );
try{
   //Send information to coordinator
this.out.write(msg);

}catch(Exception e){
   System.out.println("Error al mandar al coord ");
}
```

7.3.4 Servidor de direcciones

El servidor de direcciones presenta un valor añadido para una implementación real de la solución, sin embargo no es estrictamente necesario para el funcionamiento del sistema. Su función es almacenar una tabla que contiene las direcciones IP de cada una de las PAN_ID, para poder resolver las consultas de máquinas remotas.

El Servidor de direcciones abre un socket en el puerto 44444, donde espera solicitudes entrantes, el siguiente diagrama muestra el funcionamiento del servidor:

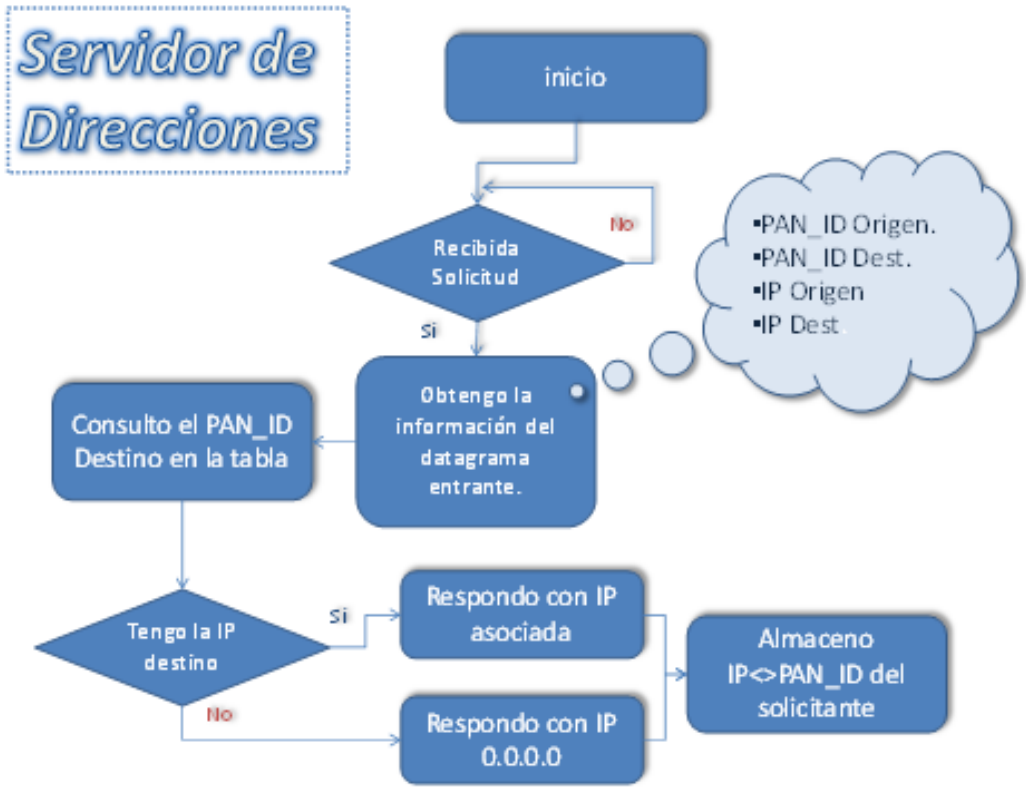
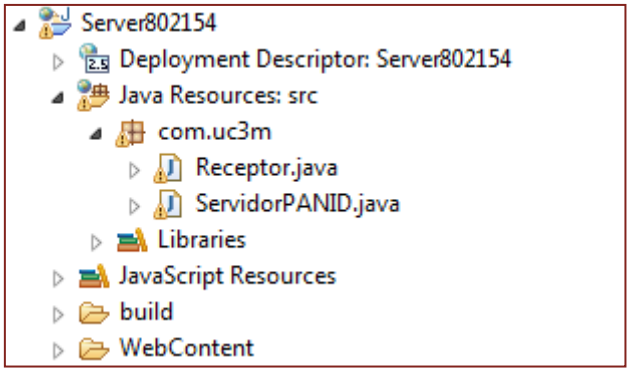


Diagrama 13: Diagrama del funcionamiento del servidor de direcciones.

7.3.4.1 *Detalle del código del servidor de direcciones*

El servidor de direcciones es una aplicación Java que es ejecutada sobre un servidor de aplicaciones, en nuestro caso Tomcat 6, consta de dos clases Receptor.java y servidorPAN_ID .java, responsables del funcionamiento de la aplicación.



La tabla con la información de direccionamiento se guarda en una HashMap Java que almacena la dupla PAN_ID<>IP, siendo la dirección de PAN_ID la clave de la relación.

```
IPTable = new HashMap<String, String>();
IPTable.put("0000DE35","192.168.1.50" );
IPTable.put("0000DE30","192.168.1.37" );
```

Cuando se recibe una solicitud, se analiza el datagrama, los 8 primeros bytes de los datos nos indican la PAN_ID sobre la cual se desea realizar la consulta.

```
String received = new String(recibido.getData());
PAN_IDRequest = received.substring(0, 8);
```

De esta forma, resulta muy sencillo realizar una búsqueda sobre la tabla.

```
try {
   String aux;
   if(IPTable.containsKey(PAN_IDRequest)){
      aux = IPTable.get(PAN_IDRequest);
   }else{
      System.out.println("Esta IP No la tengo  :(  ");
      aux = NO_IP_MSG;

   }
```

En el datagrama recibido también se encuentra información acerca de la PAN_ID que origina la consulta, por lo que utilizamos estos datos para actualizar la tabla.

```
if(!IPTable.containsKey(recibido.getAddress())){
   System.out.println("Esta me la aprendo..... "+ recibido.getAddress().toString()
   IPTable.put(received.substring(9, 17),recibido.getAddress().toString() );
}
```

Tras la consulta se envía la respuesta al cliente, además si la dirección no se encuentra en la tabla se envía una dirección IP no valida 0.0.0.0, que será interpretada por el cliente como destino inalcanzable.

7.4 Aplicación de la propuesta en un escenario real

Existen diferentes contextos donde la solución implementada se ajustaría perfectamente a las necesidades. Prácticamente podríamos hablar de cualquier escenario del tipo WPAN donde el área de cobertura superase las especificaciones de las redes de sensores y actuadores.

La domótica es un campo que está teniendo un gran despliegue de este tipo de redes en los últimos años, por desgracia los costes de los sistemas propietarios actualmente impiden una mayor expansión en el mercado para el gran público.

Este tipo de sistemas ofrecen sobre todo comodidades y seguridad, además las empresas buscan cada vez más una mayor eficiencia energética, por lo que la demanda de este tipo de sistemas está en constante aumento.

Este sería un ejemplo de un escenario de domótica.

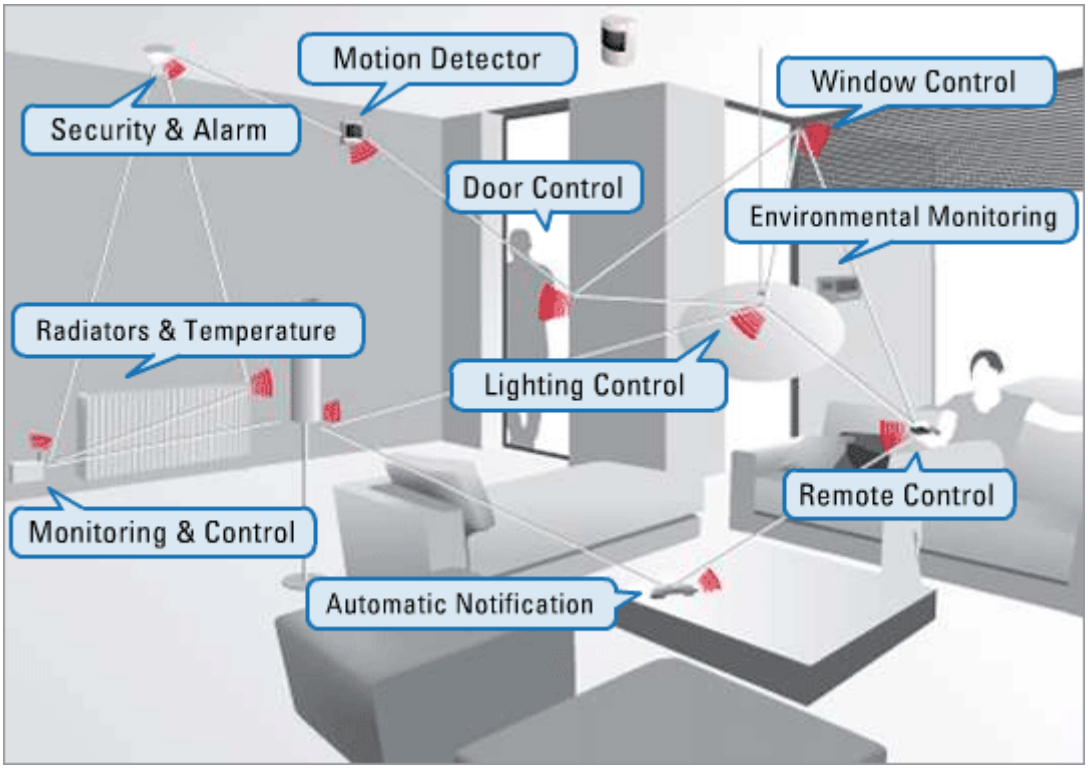


Ilustración 33: Escenario de Domótica

Como vemos en la ilustración anterior son muchas las posibilidades que nos presentan este tipo de soluciones.

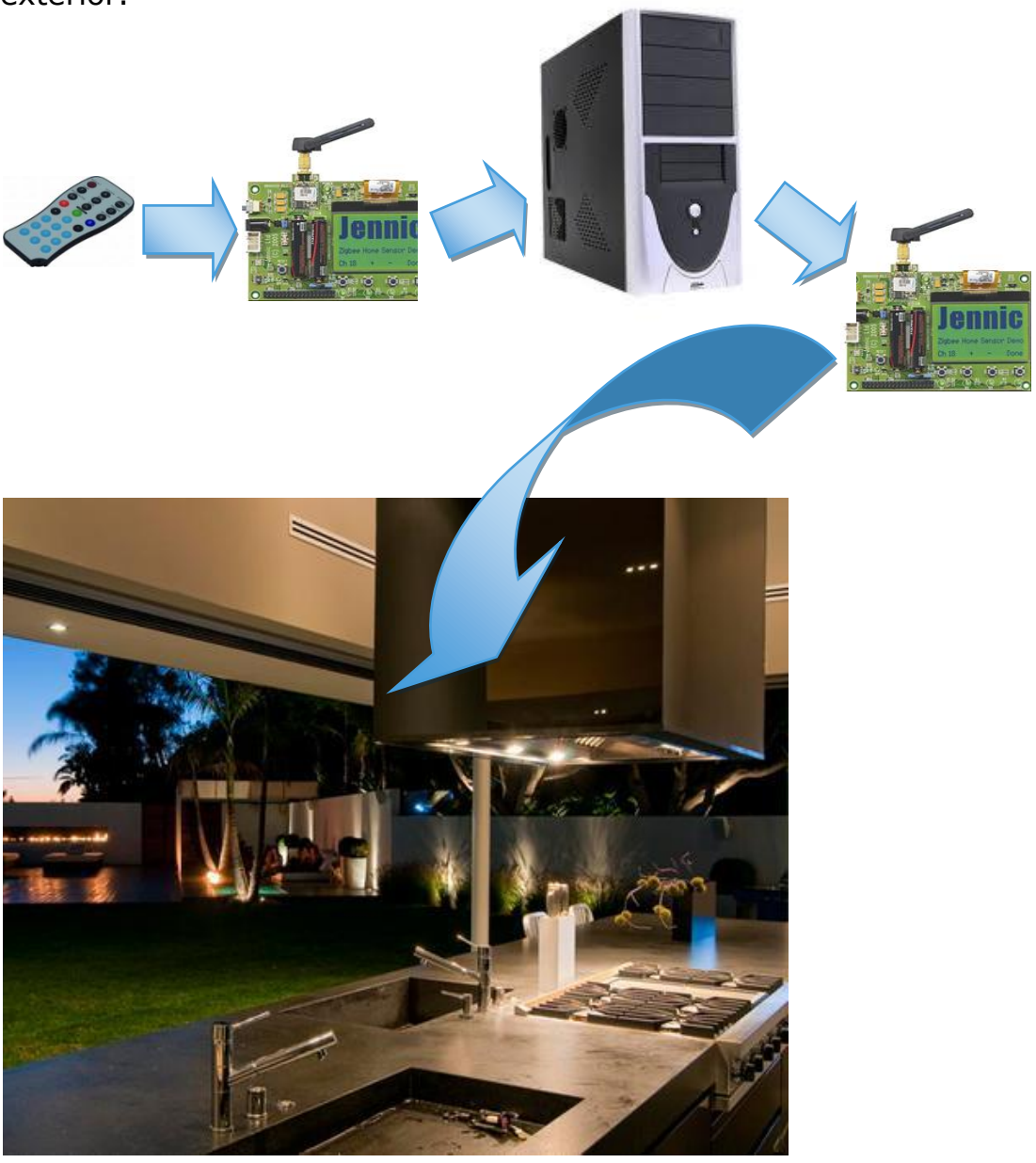
Nos Centráremos en una aplicación más concreta, para visualizar las aplicaciones que nos ofrece nuestra solución, en este caso imaginemos una vivienda de grandes dimensiones o un hotel resort, donde existen diferentes estancias separadas por un jardín, donde nuestro cliente desea tener el control de diferentes automatismos.

En este caso, sería imposible cubrir toda la vivienda con una única red de sensores, pues las distancias son demasiado grandes, y posiblemente existan atenuaciones de la señal inalámbrica debido a los materiales de construcción.

Por otra parte suponemos que la vivienda dispone de cableado UTP para conexiones a internet, o mejor aún, cuenta con enrutadores inalámbricos que cubren todo el perímetro. Pues este tipo de instalaciones son de uso prácticamente "obligado" actualmente.

Este sería un escenario idóneo para implementar nuestra solución, donde podrían desplegarse diferentes redes de sensores actuadores reguladas por sus respectivos coordinadores con acceso a la red.

Con la solución propuesta, podríamos interconectar todos los sensores y actuadores. Por ejemplo accionar la iluminación del jardín o de un quincho exterir desde un control remoto inalámbrico, en ese ejemplo concreto, las tramas 802.15.5 serian emitidas en cada pulsación del mando con destino al coordinador. Este encaminaría la trama por el puerto serie a un PC centralizado, donde se reenviaría por la red privada de datos hasta el coordinador de la red sobre la cual están sincronizados los actuadores que encienden la iluminación exterior.



8 Presupuesto.

Para la estimación contemplamos además de los costes materiales de los equipos utilizados, aspectos de los costes reales de investigación desarrollo e instalación.

8.1 Costes de investigación y desarrollo.

Los costes de investigación y desarrollo del proyecto se estiman mediante un desglose de las actividades realizadas, suponiendo un coste de personal de 200 € a la semana.

Actividad	Duración (Semanas)	Costes €/Semana
Parte 1. Estudio del proyecto	**4**	
Planteamiento de Problema y objetivos	1	
Estado del arte	2	
Definición de la propuesta	1	
Parte 2. Implementación de la Propuesta	**17**	
Familiarización con los dispositivos de Jennic	1	
Interpretación de las librerías de desarrollo, compilación y carga de aplicaciones	3	**200 €**
Desarrollo del código en los Endpoint	2	
Desarrollo del código del Coordinador	4	
Comunicación Coordinador-PC	2	
Desarrollo PCBridge	3	
Comunicación TCP/IP	1	
Test de prueba	1	
Parte 3. Documentación	**6**	
Total	**27**	**5.400 €**

Tabla 2: Desglose de las Actividades del proyecto

A continuación un diagrama de Gantt del proyecto.

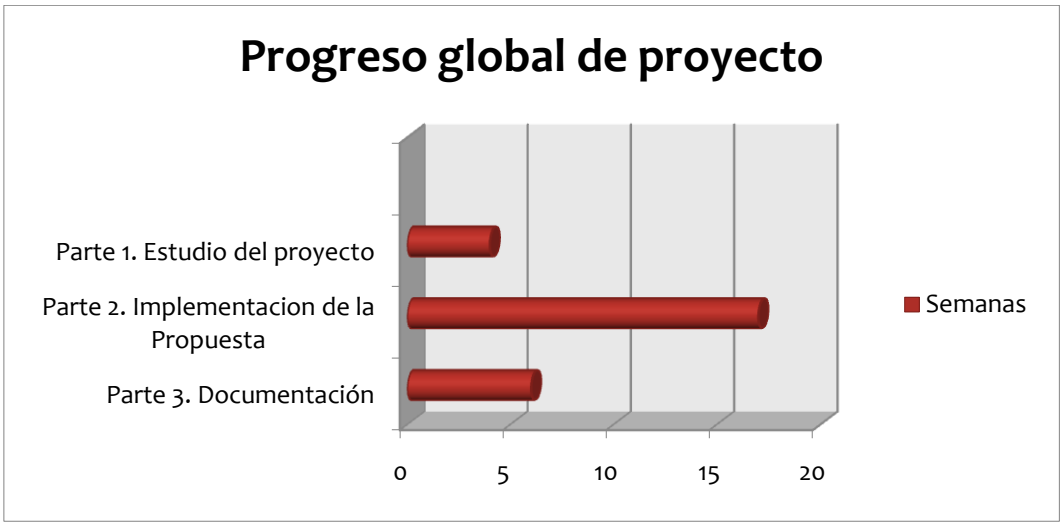


Diagrama 14: Diagrama de Gantt de desarrollo del proyecto expresado en semanas.

8.2 Costes Material

Descripción	Unidades	Coste	Total
Kit de desarrollo Jennic JN-5121	1	220 €	220 €
Placa LCD	1		
Placa Sensores	4		
Cable USB-Serie	1		
Ordenador sobremesa (CPU)	3	150 €	450 €
Cable USB-Serie	1	5 €	5 €
Total	**11**		**675 €**

8.3 Coste Total del proyecto

Total investigación y desarrollo	**5.400,00 €**
Total Material	**675,00 €**
Total Proyecto	**6.075,00 €**

8.4 Presupuesto solución domótica

Continuando el escenario mencionado anteriormente, podemos realizar una aproximación del coste que supondría la instalación del proyecto en una infraestructura domótica.

Vamos a suponer que la única necesidad de nuestro cliente es controlar la iluminación del jardín, la piscina y el quincho exterior.

8.4.1 Presupuesto solución UC3M

Para cerrar un presupuesto aproximado, debemos incluir algún controlador eléctrico con relé, que conectado a las placas de Jennic nos permita actuar sobre la iluminación. El controlador de la siguiente imagen sería suficiente.

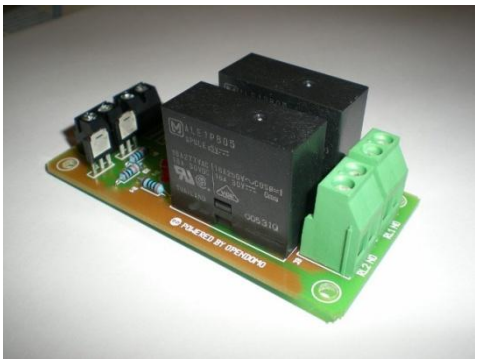


Ilustración 34: controlador eléctrico relé

Además incluimos en el presupuesto un 20% de los costes de investigación y desarrollo.

Descripción	Unidades	Coste	Total
Kit de desarrollo Jennic JN-5121	1	220 €	220 €
Placa LCD	1		
Placa Sensores	4		
Cable USB-Serie	1		
Ordenador sobremesa (CPU)	3	150 €	450 €
Cable USB-Serie	1	5 €	5 €
Controlador eléctrico 2 canales vía relé	3	20 €	60 €
Investigación y desarrollo	0,2	5.400 €	1.080 €
Total	**14,2**		**1.815 €**

8.4.2 Presupuesto solución AMX

Para una mejor perspectiva de los costes de nuestra solución, vamos a contrastarlo con un dispositivo de control AMX [9], un sistema propietario de control.

El escenario de domótica propuesto, sería cubierto con dos módulos de AMX.

MIO Modero R-4: Mando de control remoto que dispone de una pantalla LCD y 29 botones programables entre otras prestaciones. Utiliza la tecnología de comunicación ZigBee para comunicarse con la estación receptora.

Ilustración 35: MIO R4

NI-700: Controlador que dispone de puertos de entrada y salida.

Ilustración 36: NI 700

Estos dispositivos nos permitirían resolver la solución planteada.

Diagrama 15: Diagrama solución AMX

Descripción	Unidades	Coste	Total
Kit MIO Modero R-4	1	1.200 €	1.200 €
NI-700	1	822 €	822 €
Total	**2**		**2.022 €**

8.4.3 Conclusiones Presupuestos domótica.

Tras mostrar el presupuesto aproximado para las dos soluciones, podemos ver como la solución expuesta en este proyecto es unos 400€ más barata que la solución del sistema propietario AMX.

Debemos considerar que se han incluido importantes gastos de investigación y desarrollo en nuestra solución, por lo cual la diferencia podría ser aún mayor.

No obstante, aunque la solución de AMX nos ofrece un control mucho más atractivo para el usuario, existen dos factores que consideramos de vital importancia que resaltarían las prestaciones de nuestra solución:

- Los dispositivos inalámbricos de AMX necesitan ser recargados con frecuencia (unas 2 veces a la semana), mientras que los dispositivos con la solución UC3M pueden permanecer largos periodos sin recarga.

- La solución propuesta es una medida global, capaz de implementarse en escenarios donde las WPAN estén separadas tanto como sea necesario. El sistema de AMX puede también integrarse totalmente con la red TCP/IP para interactuar de forma remota, aunque esto supondría elevados costes añadidos a los presupuestados.

9 Tutorial de despliegue del ejemplo

En este apartado se describirán los pasos necesarios para la instalación y puesta en marcha el escenario implementado.

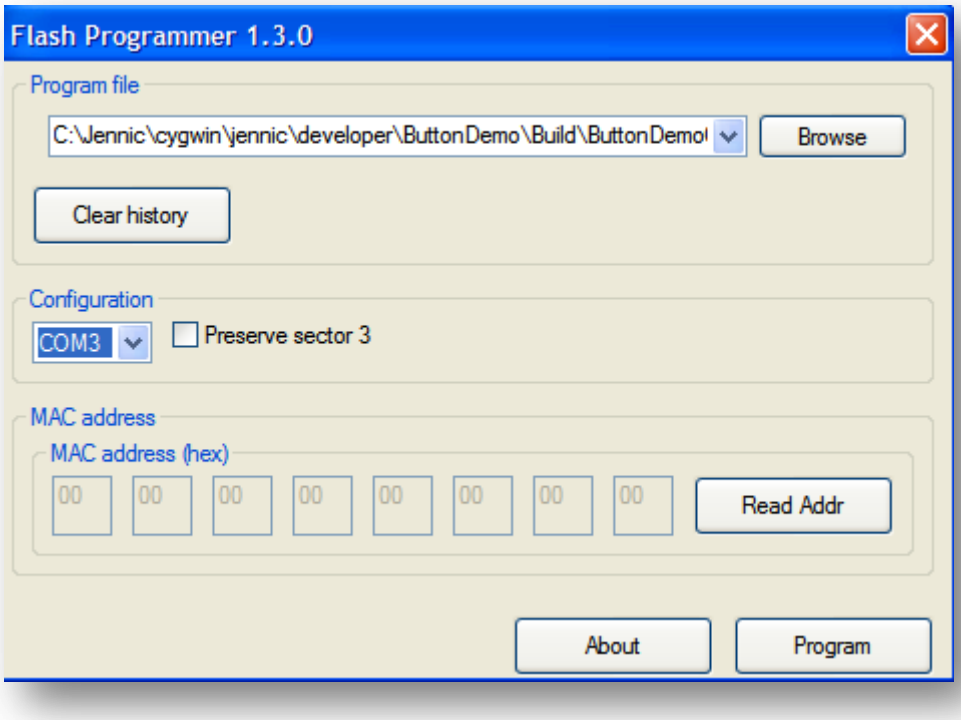
Paso 1: El material necesario para el despliegue es el siguiente.

- 2 ordenadores con Windows instalado, puerto serie, y conexión a internet o red local.
- Kit de desarrollo Jennic JN 5121..
- Cable RS232<->USB (En el caso de no disponer de Puerto serie)

Paso 2: Empezaremos programando los dispositivos de Jennic, para ello utilizamos un software especifico que nos permitirá cargar los programas compilados en la memoria flash.

El programa se llama "J5121Flash Programmer", se encuentra en la carpeta de Jennic software del proyecto, la instalación es bastante sencilla y requiere de la instalación de otros programas como cygwin que se instalaran de forma automática.

Una vez instalada la aplicación, podemos cargar los archivos compilados (.BIN), para ello arrancamos la aplicación "Flash programmer", apagamos el dispositivo y lo conectamos con el puerto serie, con la aplicación arrancada encendemos el dispositivo y presionamos sobre "Read Addr", esto nos debería rellenar los campos con la dirección MAC. A continuación seleccionamos el archivo .BIN que deseamos cargar y presionamos "Program".

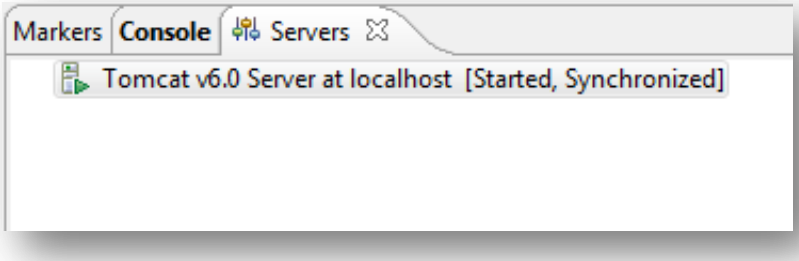


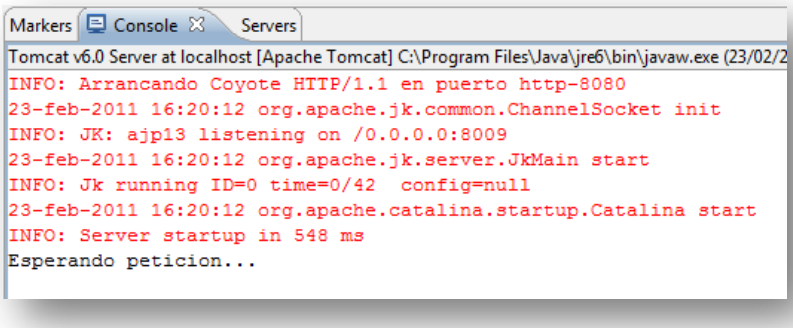
Paso 3: Instalación de eclipse, podemos encontrar en internet diferentes distribuciones, todas ellas serán validas para el desarrollo del ejemplo. Una vez instalado podemos importar los proyectos de PCBridge y Server802154, ambos se encuentran en la carpeta "Java" del proyecto.

Paso 4: Para que la aplicación Java interprete los datos recibidos por el puerto serie, se utiliza una librería de código abierto llamada RxTx, para su instalación podemos seguir el manual que se encuentra en la carpeta RxTx del proyecto, contiene toda la información y librerías necesarias.

Nota: recordamos que el flujo de datos del ejemplo es simétrico en ambos sentidos, por lo cual debemos instalar las aplicaciones en los dos ordenadores

Paso 5: El servidor de direcciones puede estar instalado en una maquina remota, o bien en una de las dos maquinas conectadas a las WSN, en cualquier caso es necesario un servidor de aplicaciones, puede utilizarse cualquiera, en nuestro caso hemos utilizado Apache Tomcat, el cual puede ser arrancado desde eclipse, lo que resulta muy cómodo, en la carpeta "instalacion Tomcat" se encuentra el manual de instalación y los archivos necesarios. Si la instalación se ha realizado con éxito podremos importar el proyecto en eclipse, y ejecutarlo seleccionando "run on server".





Paso 6: Antes de ejecutar el servidor de direcciones debemos rellenar su tabla en el código Java de la aplicación. Los valores por defecto son los siguientes:

```
//Tabla de direcciones
IPTable.put("0000DE35","192.168.1.37" );
IPTable.put("0000DE30","192.168.1.50" );
```

Además debemos indicar a los PCBridge donde está localizado el servidor de direcciones. En el ejemplo siguiente está situado en la propia maquina.

```
//Direccion IP del servidor de PAN_ID
String iPDestination = "127.0.0.1";
```

10 Conclusiones

10.1 Evaluación de resultados

El estudio precedente al desarrollo de la propuesta nos planteo una serie de metas las cuales han sido alcanzadas con éxito, tras la implementación podemos considerar como adecuada la solución establecida.

Los dispositivos de Jennic en los Endpoint mantienen una comunicación totalmente transparente para ellos, lo que implica poder simplificar al máximo sus procedimientos consiguiendo así las premisas de las redes WSAN de bajo coste y reducir el consumo.

Los dispositivos de Jennic en los Coordinadores establecen una rápida interconexión de los dispositivos con tiempos prácticamente despreciables. La comunicación de los coordinadores con los PC es 100% fiable mediante el puerto serie utilizado.

Podemos reflejar también que en un escenario real, no todos los dispositivos de una WSAN, por su naturaleza, necesitan tener comunicación fuera de la red, la solución propuesta permite combinar de forma transparente para los Endpoint dispositivos de los dos tipos.

La aplicación PCBridge.java, consigue eficientemente reenviar los paquetes que se reciben por el puerto serie, la elección de una aplicación Java multi-hilo resuelve sobradamente las necesidades de intercomunicación del flujo de datos. Quizás como contrapartida hubiera sido interesante desarrollar alguna interfaz gráfica de usuario para la aplicación, de forma no solo podamos visualizar la línea de comandos, otorgando a la aplicación de un mayor atractivo de cara al usuario.

10.2 Trabajos futuros

Este proyecto a cumplido el objetivo de intercomunicar redes de sensores remotas, donde se han integrado 802.15.4 y TCP/IP.

Este proyecto abre posibilidades para plantear algunos trabajos futuros.

- En primer lugar sería interesante aplicar el proyecto a un escenario de domótica como el planteado. Donde los dispositivos se comuniquen desde diferentes zonas.
  Los dispositivos de Jennic con los que se ha desarrollado el ejemplo, disponen de sensores de luz, temperatura y humedad, además, uno de los dispositivos incorpora una pantalla LCD. Inicialmente se realizaron pruebas para comprobar el funcionamiento de dichos sensores y de la pantalla LCD, no continuamos trabajando en este campo pues no era el objetivo que se planteaba en el proyecto.
  En un futuro podría trabajarse sobre este escenario intercambiando los datos recogidos por los sensores.

- Utilizar el estándar de ZigBee sobre este mismo proyecto, modificando tanto el código de los Endpoint como el del coordinador, esto daría mayor robustez al sistema, además de abrir la puerta a otros campos de investigación como puede ser el multisalto.

- Otro factor que no se ha trabajado en el proyecto es la cuestión de seguridad, sería interesante investigar acerca de la seguridad en 802.15.4. Al mismo tiempo se podría plantear fácilmente el filtrado de las tramas por parte de los PC, implementando tablas de permisos a los Endpoint, de esta forma podríamos controlar por filtrado MAC la comunicación de los dispositivos.

11 Referencias

[0]A Proposal for ZigBee Clusters Interconnection based on Zigbee Extension Devices
Angel Cuevas, Ruben Cuevas, Manuel Urueña, and David Larrabeiti
Departamento Ingenieria Telematica. Universidad Carlos III de Madrid

[1]SENSORES Y TRANSDUCTORES.
http://www.profesormolina.com.ar/tecnologia/sens_transduct/index.htm

[2]Aplicaciones desarrolladas por la empresa española Libelium.
Destacada internacionalmente por sus aplicaciones en WSN.

[3]Redes de Sensores Inalámbricos.
Las tecnologías de la información y de las comunicaciones en el ámbito de la atención socio sanitario.
Francisco Gómez Mula
Dpto. de Arquitectura y Tecnología de Computadores
Universidad de Granada

[4]Redes de Sensores y Actuadores (WSAN)
Manuel J. Buendía, Jose A. Vera, Fernando Losilla, Pedro José Meseguer
DSIE, Universidad Politécnica de Cartagena, Campus Muralla del Mar s/n., 30202, Cartagena, SPAIN

[5]IEEE Computer Society
Part 15.4: Wireless Medium AccessControl (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)

[5]802.15.4 y Zigbee
Jordi Mayné Ingeniero de Aplicaciones ; Freescale semiconductor; Silica

[6]Zigbee: "Wireless Control That Simply Works"
William C. Craig Program Manager Wireless Communications; ZMD America, Inc.

[7]ZigBee Alliance specification
27 Junio 2005

[8]Comunicaciones Inalámbricas
Ivan Bernal PhD ; Escuela politécnica nacional de Quito

[9]AMX IBERIA
http://amx.com.es/index.php

[10]JENNIC TECHNOLOGY FOR CHANGING WORLD
http://www.jennic.com/